NEURAL BARRIER FUNCTIONS AS GUARDRAILS FOR ADAPTIVE CHAIN-OF-THOUGHT STEERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural barrier functions (NBFs) provide a principled way to certify probabilistic safety and reachability in stochastic systems. Leveraging the inherent stochasticity of Large Language Model (LLM) generation, we introduce GUARDSTR, a framework that formulates Chain-of-Thought (CoT) reasoning as a finite-horizon stochastic reachability problem, where successful reasoning corresponds to reaching latent states of correct final answers. Based on this formulation, we learn a *reasoning barrier function* as a guardrail to steer CoT reasoning. Theoretically, it provides probabilistic guarantees under a set of principled conditions for the desired terminal outcomes. Practically, it assigns a quantitative feedback credit for adaptive activation steering at every reasoning step. GUARDSTR requires neither parameter fine-tuning nor ground-truth answers during inference, ensuring compatibility with different open-source LLMs. Across four LLMs and seven tasks from BBH and MMLU, GUARDSTR improves final-answer accuracy over unsteered baselines on all 28 model–task pairs, yielding a macro-average gain of 5.91 percentage points. It achieves maximum gains of 12.8 and 11.6 percentage points over two state-of-the-art methods, Static CAA and K-CAST, respectively.

1 INTRODUCTION

Chain-of-thought (CoT) reasoning enables large language models (LLMs) to tackle complex problems through sequences of intermediate reasoning steps (Wei et al., 2022; Kojima et al., 2022). In long-horizon tasks, an early error can affect subsequent decisions, and locally plausible steps can still lead to an incorrect final answer. Activation steering offers a way to guide this process at inference time by modifying internal representations without updating the base model's parameters (Rimsky et al., 2024). For adaptive steering, a central question is: *what feedback should guide these interventions to help the model reach a correct final answer within a finite reasoning budget?*

Existing work uses behavioral contrasts to estimate steering directions (Turner et al., 2023; Zou et al., 2023; Rimsky et al., 2024) and activation-based signals to condition the application of interventions (Lee et al., 2025). Recent work further views CoT as an evolving trajectory in representation space and develops interventions at individual reasoning steps (Sun et al., 2026; Valentino et al., 2026). The empirical effectiveness of these interventions, however, does not by itself establish conditions connecting the controlled reasoning process to task success within a finite reasoning budget. We therefore seek a *guardrail* that supplies this connection: explicit conditions whose satisfaction supports guarantees on reaching a correct final answer within the reasoning budget, together with feedback for evaluating and guiding step-level interventions.

Stochastic barrier certificates provide a foundation for such guardrails: conditions on a scalar state function can establish bounds on trajectory-level success probabilities (Prajna et al., 2007; Salamati & Zamani, 2022; Zhi et al., 2024). Building on this principle and barrier-based LLM representation steering (Tran et al., 2026), we formulate CoT reasoning as a finite-horizon stochastic reachability problem over latent representation states. These states are extracted from the LLM's hidden-layer activations at successive reasoning steps, with next-step sampling inducing stochastic transitions between them. We define a *reasoning barrier function* on these states and use it as a guardrail for credit assignment in GUARDSTR. Theoretically, when its task-related boundary and stochastic transition conditions hold for the controlled process, the function yields a lower bound on the probability of reaching a correct final answer within the reasoning budget. Practically, we learn the

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.

function from outcome-labeled trajectories without intermediate-step annotations and use it to assign transition-level credit and guide state-dependent gradient interventions in response to condition violations. These interventions require no fine-tuning of the base LLM and no ground-truth answers at inference time. The barrier function thus connects step-level credit assignment and adaptive CoT steering to final-answer correctness through explicit conditions on the controlled reasoning process.

We evaluate GUARDSTR on four LLMs across seven reasoning tasks from BBH and MMLU (Suzgun et al., 2023; Hendrycks et al., 2021). It improves final-answer accuracy over the corresponding unsteered models on all 28 model–task pairs, achieving a macro-average gain of 5.91 percentage points. The largest improvements on model–task pairs over two state-of-the-art methods Static CAA (Rimsky et al., 2024) and K-CAST (Valentino et al., 2026) are 12.8 and 11.6 percentage points, respectively. These results demonstrate the empirical effectiveness of barrier-guardrail intervention, complementing the conditional theoretical guarantees. Our main contributions are as follows:

- We introduce *guardrail steering*, which grounds activation interventions in explicit, task-related certificate conditions. We establish how satisfaction of these conditions yields a bound on the probability of reaching a correct answer within the reasoning budget.
- We propose a neural approach to *learning guardrails* from outcome-labeled reasoning trajectories without intermediate-step annotations. A learned guardrail supports both offline transition-level credit assignment and online violation-driven steering through state-dependent gradient interventions.
- We implement GUARDSTR and evaluate it on four LLMs across seven reasoning tasks, demonstrating consistent accuracy improvements over unsteered models and gains over existing activation-steering baselines.

2 RELATED WORK

Adaptive Chain-of-Thought Steering. Activation-steering methods estimate intervention directions from behavioral contrasts (Turner et al., 2023; Zou et al., 2023; Rimsky et al., 2024), while CAST conditions interventions on activation-based signals (Lee et al., 2025). Recent work studies CoT as an evolving trajectory in representation space and develops interventions at individual reasoning steps (Sun et al., 2026; Valentino et al., 2026). These approaches draw on several sources of steering feedback. DynaSteer constructs contrastive labels at reasoning forks, while Future Probe Controlled Generation predicts future behaviors to guide sentence selection (Wang et al., 2026; Kortukov et al., 2026). Process supervision provides another source of feedback: ATLAS distills process reward scores into a latent verifier, whereas GeoSteer learns a differentiable quality function from teacher-scored prefixes (Nguyen & Le, 2026; Kazama et al., 2026). Meta-Reasoner and Cognitive Demand Steering use LLM assessments of reasoning progress and remaining cognitive demands, respectively (Sui et al., 2026; Scoville et al., 2026). GUARDSTR uses transition residuals of barrier conditions as feedback for adaptive activation steering.

Barrier Functions for Safety and Reachability. Barrier functions connect local conditions on system dynamics to trajectory-level safety and reachability bounds (Prajna et al., 2007; Xue, 2025). Neural parameterizations allow such certificates to be learned from data for complex systems. Neural Lyapunov-barrier functions support safe nonlinear robot control under model uncertainty (Dawson et al., 2021), while decentralized neural barrier certificates enable multi-agent navigation with obstacle and inter-agent collision avoidance (Qin et al., 2021). Neural certificates also support qualitative and quantitative safety verification of DNN-controlled systems under stochastic perturbations (Zhi et al., 2024). For language generation, BarrierSteer learns latent-space control barrier functions and uses their constraints to address unsafe outputs, including harmful generation and adversarially induced behaviors (Tran et al., 2026). GUARDSTR targets finite-horizon correct-answer reachability, with a lower bound on success probability conditional on the certificate conditions holding for the controlled process.

3 COT REASONING AS STOCHASTIC REACHABILITY

CoT generation produces stochastic reasoning rollouts for the same problem. For the date question in Figure 1, one rollout correctly identifies today's date, while another instead computes Jane's

appointment date. Representations extracted at successive reasoning steps from an intermediate layer ℓ form a latent state trajectory for each rollout. The successful trajectory (green) reaches the correct-answer target set within the reasoning budget, whereas the failed trajectory (red) does not.

To characterize the probability of reaching this target, we first formalize CoT reasoning as a finite-horizon stochastic reachability problem. We then introduce reasoning barrier functions whose boundary and transition conditions yield a lower bound on this probability. Finally, we develop a neural learning objective that encourages satisfaction of these conditions.

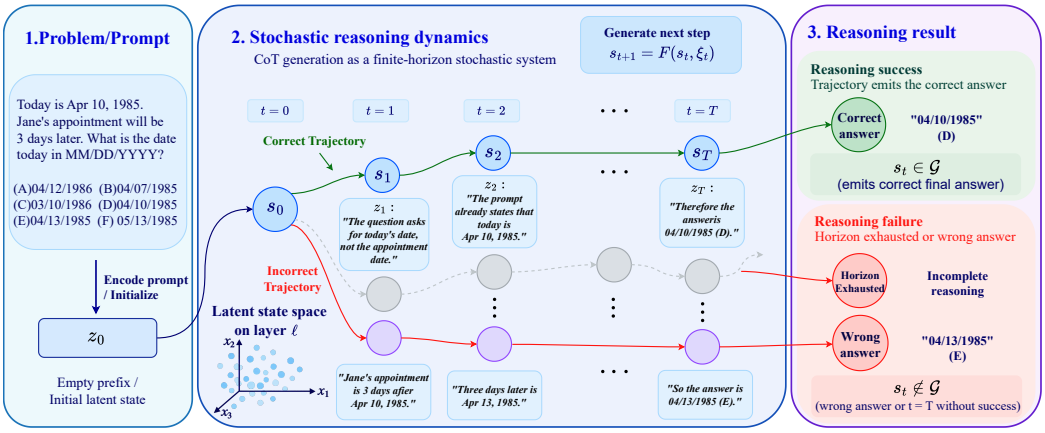

Figure 1: CoT Reasoning as a Finite-Horizon Stochastic Reachability Problem

3.1 Formalizing CoT Reasoning

Given a problem x as input prompt, an LLM π with a fixed decoding configuration, and a reasoning budget $T_{\max} \in \mathbb{N}_{>0}$, we sample a rollout $\kappa \sim \pi(\cdot \mid x)$ containing the generated reasoning and any final response. Following Sun et al. (2026), we partition κ using explicit "Step t: z_t" markers, where z_t denotes the content of the t-th segment. The resulting sequence $(z_1, \ldots, z_T)$ forms a reasoning trajectory, with $T \le T_{\max}$ segments generated before termination. A stopping event closes the final segment, which may contain a final answer or a truncated reasoning step.

To represent this trajectory in latent space, we select an intermediate layer ℓ of π with hidden dimension d_ℓ. At each step t, let $z_{\le t} = (x, z_1, \ldots, z_t)$ denote the complete prefix, including the problem and all reasoning segments generated so far. A fixed representation-extraction map h^ℓ maps this prefix to the layer-ℓ representation of the current reasoning step:

$$s_t = h^\ell(z_{\le t}) \in \mathbb{R}^{d_\ell}, \qquad t \in [1, T].$$

Applying the same map at every step yields the latent state trajectory $\tau = (s_1, \ldots, s_T)$. Let $\mathcal{S} \subseteq \mathbb{R}^{d_\ell}$ denote the space of admissible latent states.

Reasoning stochastic process. The latent trajectory evolves through successive prefix extensions. Given a nonterminal prefix $z_{\le t}$, the LLM samples the next segment $z_{t+1} \sim \pi(\cdot \mid z_{\le t})$ and appends it to obtain $z_{\le t+1}$. Let ξ_t denote the fresh sampling randomness used to generate this segment under the fixed decoding rule, such as the draws in temperature-based sampling. The induced latent transition can then be written as

$$s_{t+1} = F(s_t, \xi_t) = h^\ell(z_{\le t+1}), \qquad t = 1, \ldots, T-1,$$

where F combines the autoregressive generation of one reasoning segment with representation extraction from the extended prefix. This construction defines a *reasoning stochastic process* $\mathcal{M}$ on $\mathcal{S}$ induced by the LLM's prefix-conditioned generation.

Correct-Answer Reachability. For a fixed problem x, let $\mathcal{A}_x$ denote the set of accepted correct final answers. Let $\mathcal{P}^x_{\text{term}}$ contain all possible complete histories $p = (x, z_1, \ldots, z_T)$, where the random termination step $T \le T_{\max}$ is determined by the stopping rule or budget exhaustion. For each $p \in \mathcal{P}^x_{\text{term}}$, $\text{Ans}(p)$ parses the answer associated with the last "Final answer:" marker in the

generated text according to the prescribed answer format. An absent marker or an unparseable answer yields $\perp \notin \mathcal{A}_x$. We define the *latent target set* as

$$\mathcal{G}_x = \left\{ h^\ell(p) \;\middle|\; p \in \mathcal{P}^x_{\text{term}},\ \text{Ans}(p) \in \mathcal{A}_x \right\} \subseteq \mathcal{S}.$$

To identify target membership with correct termination, we require that no nonterminal or incorrectly terminated prefix maps into $\mathcal{G}_x$.

For analysis, we extend each trajectory by setting $s_t = s_T$ for $T < t \leq T_{\max}$. Under the correspondence above, the finite-horizon success probability is

$$p_{\text{succ}} = \Pr(\exists\, t \in \{1, \ldots, T_{\max}\} : s_t \in \mathcal{G}_x) = \Pr(\text{Ans}(z_{\leq T}) \in \mathcal{A}_x)\,.$$

The probability is taken over generation randomness under the specified reasoning process. Termination without an accepted correct final answer, including budget exhaustion, constitutes failure.

Problem Formulation. We seek a *guardrail* for adaptive steering, defined as a set of conditions on the controlled reasoning process. When satisfied, these conditions should certify a lower bound on the probability of correct termination within the reasoning budget:

$$p_{\text{succ}} \geq \underline{p}, \qquad \underline{p} \in [0, 1].$$

The guardrail should also provide a criterion for evaluating individual reasoning transitions, with violations supplying feedback for steering step-level interventions toward correct termination. The next section introduces *reasoning barrier functions* to formulate these conditions and derive the associated reachability bound.

3.2 Reasoing Barrier Functions

Barrier functions connect local conditions on a scalar function to trajectory-level probabilistic guarantees (Prajna et al., 2007; Salamati & Zamani, 2022). Inspired by this principle, we introduce a *reasoning barrier function* whose boundary and transition conditions yield a lower bound on correct-answer reachability.

For a fixed problem x, let $T \leq T_{\max}$ denote the termination step and $\mathcal{G}_x$ the target set defined above. We extend the indexing by setting $z_{\leq 0} = x$ and $s_0 = h^\ell(x)$. For analysis, we hold the prefix and its latent representation fixed after termination: $z_{\leq t} = z_{\leq T}$ and $s_t = s_T$ for $t > T$.

Definition 1 (Reasoning Barrier Functions)**.** *A measurable function $B_\phi : \mathcal{S} \to \mathbb{R}_{\geq 0}$ is a reasoning barrier function for the reasoning stochastic process $\mathcal{M}$ if there exist constants $\beta \in (0, 1)$, $\beta < \alpha < 1 + \beta$, and $\epsilon \geq 0$ such that*

$$B_\phi(s) \leq \beta, \qquad \forall s \in \mathcal{S} \setminus \mathcal{G}_x, \tag{1}$$

$$\alpha \leq B_\phi(s) \leq 1 + \beta, \qquad \forall s \in \mathcal{G}_x, \tag{2}$$

$$\mathbb{E}[B_\phi(s_{t+1}) \mid s_t = s] \geq B_\phi(s) + \epsilon, \qquad \forall s \in \mathcal{S} \setminus \mathcal{G}_x. \tag{3}$$

The boundary conditions (Eqs. (1),(2)) separate correct terminal states from the remaining states by a margin of at least $\alpha - \beta$. The transition condition (Eq. (3)) requires an expected increase of at least ϵ at each step before termination; after termination, it holds with equality. All expectations are taken under the reasoning process being analyzed, including its interventions when steering is applied.

Proposition 1 (Finite-Horizon Correct-Answer Reachability)**.** *Suppose B_ϕ satisfies Definition 1. Under the correspondence between target membership and successful termination specified above, the success probability satisfies that for any trajectory starting from s_0,*

$$p_{\text{succ}} = \Pr(\exists\, t \in \{1, \ldots, T_{\max}\} : s_t \in \mathcal{G}_x) \geq \max\left\{0,\ B_\phi(s_0) + \frac{1}{2}\epsilon T_{\max} - \beta\right\}. \tag{4}$$

This bound connects expected progress across reasoning steps to the probability of a correct final answer. It applies when the barrier conditions hold for the reasoning process under consideration. The proof is given in Appendix A.

3.3 Learning Neural Barrier Functions

We parameterize B_ϕ with a neural network to learn a scalar function over high-dimensional LLM representations and obtain the state gradients needed for activation steering. A nonnegative output enforces $B_\phi(s) \geq 0$ by construction. We then train the network using three hinge penalties corresponding to the boundary and transition conditions in Definition 1.

Let $[u]_+ = \max\{u, 0\}$. The first loss penalizes violations of the upper bound on non-target states:

$$\mathcal{L}_1(\phi) = \sup_{s \in \mathcal{S} \setminus \mathcal{G}_x} [B_\phi(s) - \beta]_+. \tag{5}$$

The second loss penalizes target-state values outside the interval $[\alpha, 1 + \beta]$:

$$\mathcal{L}_2(\phi) = \sup_{s \in \mathcal{G}_x} \left([\alpha - B_\phi(s)]_+ + [B_\phi(s) - (1 + \beta)]_+ \right). \tag{6}$$

Together, these losses encourage the separation between target and non-target states required by Eqs. (1) and (2).

The third loss addresses the expected-progress condition in Eq. (3). Because the conditional expectation is not directly available, we use a transition-level surrogate that penalizes individual barrier increments below ϵ. Let $\mathcal{E}_x \subseteq (\mathcal{S} \setminus \mathcal{G}_x) \times \mathcal{S}$ denote the set of admissible one-step latent transitions under the reasoning process. We define

$$\mathcal{L}_3(\phi) = \sup_{(s,s') \in \mathcal{E}_x} [B_\phi(s) + \epsilon - B_\phi(s')]_+. \tag{7}$$

This penalty encourages progress on individual transitions and serves as a surrogate for the conditional-expectation constraint.

We minimize the weighted objective

$$\mathcal{L}_{\text{full}}(\phi) = \lambda_1 \mathcal{L}_1(\phi) + \lambda_2 \mathcal{L}_2(\phi) + \lambda_3 \mathcal{L}_3(\phi), \tag{8}$$

where $\lambda_1, \lambda_2, \lambda_3 > 0$ balance the boundary and transition penalties. In practice, we train B_ϕ using states and transitions sampled from outcome-labeled reasoning trajectories. Correct terminal states receive target labels and all other states receive non-target labels, without requiring intermediate-step correctness annotations.

4 Guardrail Steering via Reasoning Barrier Functions

4.1 Overview of Guardrail Steering

We use the reasoning barrier function in Section 3.3 to implement the guardrail for online CoT steering. Since training reduces the boundary-condition violations (Eqs. (1) and (2)) to negligible levels, online evaluation treats the **violation of the transition condition** (Eq. (3)) as the **guardrail**.

Figure 2 summarizes the procedure. In **(a)**, the guardrail evaluates the current CoT step at inference time, producing normalized shortfall and surplus signals (Section 4.2). The shortfall signal guides steering strength in **(b)** and a gradient direction for reducing violations in **(c)** (Section 4.3). In **(d)**, we apply the resulting activation update and resume decoding (Section 4.4). This cycle repeats until CoT generation terminates or the reasoning budget is exhausted.

4.2 Inference-time Evaluation with Guardrails

At inference time, we adapt the transition condition in Eq. (3) by using the observed next-state barrier value in place of its conditional expectation. For each generated CoT step z_t, we evaluate the latent transition at the intermediate layer ℓ from s_{t-1} to $s_t = h^\ell(z_{\leq t})$, with $s_0 = h^\ell(x)$. These states are recorded from the ongoing execution and include preceding interventions. Specifically, the resulting guardrail residual is defined according to the violation of the transition condition of the reasoning barrier function and its format is given as:

$$\Delta_t^B = B_\phi(s_{t-1}) + \epsilon - B_\phi(s_t).$$

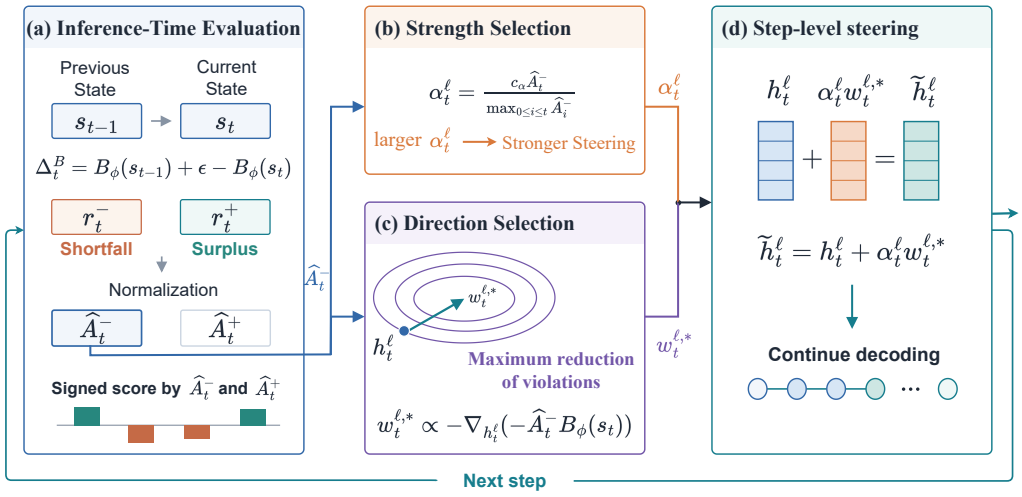

Figure 2: Pipeline of online guardrail steering

We decompose this guardrail feedback into shortfall and surplus:

$$r_t^- = [\Delta_t^B]_+, \qquad r_t^+ = [-\Delta_t^B]_+.$$

The shortfall r_t^- measures how much the current transition violates the transition barrier condition: the amount by which its barrier increment falls below ϵ. The surplus r_t^+ measures how much the increment exceeds ϵ, beyond what is required to satisfy the condition.

We normalize the two scores separately over the steps observed so far:

$$\widehat{A}_t^- = \frac{\exp(r_t^-/\tau_c)}{\sum_{j=1}^t \exp(r_j^-/\tau_c)}, \qquad \widehat{A}_t^+ = \frac{\exp(r_t^+/\tau_c)}{\sum_{j=1}^t \exp(r_j^+/\tau_c)}, \tag{9}$$

where $\tau_c > 0$ is a temperature parameter. Both values are positive and express the current step's shortfall or surplus relative to its respective history. We refer to A_t^- as the steer credit assigned to the current reasoning step and use the steer credit guide steering strength and direction in Section 4.3.

4.3 Guardrail-Guided Strength and Direction Selection

The inference-time evaluation provided by the guardrail guides both how strongly to intervene and which direction to follow. We use the steer credit $\widehat{A}_t^-$ to determine the intervention strength and the barrier gradient of reducing $\widehat{A}_t^-$ to identify a state-dependent correction direction.

Strength selection. Fix a hidden layer $\ell \in \mathcal{L}$, we determine the intervention strength using the current steer credit $\widehat{A}_t^-$ and its recorded history. We initialize $\widehat{A}_0^- = 0$ as a reference and define

$$\alpha_t^\ell = \frac{c_\alpha\, \widehat{A}_t^-}{\max_{0\le i\le t} \widehat{A}_i^-}, \tag{10}$$

where $c_\alpha \ge 0$ is a global strength coefficient. For a fixed history range, a larger steer credit produces a stronger intervention, allowing the guardrail's evaluation of the current step to control the correction strength. Particularly $c_\alpha = 0$ disables steering.

Direction selection. We select the intervention direction to locally reduce the sampled transition residual. With the preceding barrier value held fixed, increasing $B_\phi(s_t)$ decreases Δ_t^B. We therefore minimize the surrogate $J_t(s_t) = -\widehat{A}_t^- B_\phi(s_t)$, treating $\widehat{A}_t^-$ as a detached scalar and differentiating through the dependence of s_t on h_t^ℓ. Viewing J_t as a function of the selected hidden representation, a small perturbation w^ℓ gives the first-order approximation

$$J_t(h_t^\ell + w^\ell) \approx J_t(h_t^\ell) + \left\langle \nabla_{h_t^\ell} J_t(s_t), w^\ell \right\rangle.$$

For a fixed ℓ_2 perturbation magnitude, this approximation decreases most along the negative gradient. We thus use the numerically stabilized direction

$$w_t^{\ell,*} = -\frac{\nabla_{h_t^\ell} J_t(s_t)}{\left\|\nabla_{h_t^\ell} J_t(s_t)\right\|_2 + \nu}, \qquad \ell \in \mathcal{L}, \tag{11}$$

where $\nu > 0$ stabilizes normalization and yields a zero direction when the gradient vanishes.

4.4 Closed-Loop Guardrail Steering

We initialize the guardrail from the prompt and generate the first reasoning segment. At each nonterminal step boundary, the guardrail evaluates the current transition and combines the steering strength and direction from Section 4.3 to update the selected hidden representations:

$$\widetilde{h}_t^\ell = h_t^\ell + \alpha_t^\ell w_t^{\ell,*}, \qquad \ell \in \mathcal{L}. \tag{12}$$

Both quantities are computed before the update, which is applied through the corresponding activation hooks. Decoding then continues to generate the next reasoning step.

We retain the pre-intervention value $B_\phi(s_t)$ as the reference for the next transition residual. The next state s_{t+1} is recorded from the ongoing controlled execution and therefore reflects preceding interventions. Evaluating this transition provides feedback for the next correction, closing the loop between intervention and subsequent generation.

The parameters of π and B_ϕ remain fixed throughout. The loop ends when generation terminates or the reasoning budget is exhausted, after which the final answer is extracted according to Section 3.1.

5 Experimental Evaluation

We evaluate GuardSTR on four base LLMs across seven reasoning tasks. We first describe the experimental setup and NBF learning. We then compare final-answer accuracy against unsteered baselines and two state-of-the-art steering methods, and ablate adaptive steering strength. Additional results appear in Appendix B.

5.1 Experimental Setup

Benchmarks and tasks. We evaluate on seven tasks from BIG-Bench Hard (BBH) (Suzgun et al., 2023) and Massive Multitask Language Understanding (MMLU) (Hendrycks et al., 2021). The five BBH tasks assess complementary reasoning skills: *Boolean Expressions* involves evaluating Boolean operators; *Date Understanding*, temporal and calendar reasoning; *Dyck Languages*, completing balanced bracket sequences; *Geometric Shapes*, identifying shapes from drawing instructions; and *Object Counting*, counting objects described in natural language. The two MMLU subjects, *High School Physics* and *High School Mathematics*, assess physics reasoning and mathematical problem solving, respectively.

Models and baselines. We use four open-weight LLMs in the 7B–9B parameter range: Qwen3.5-9B (Qwen Team, 2026), Llama-3.1-8B-Instruct (Grattafiori et al., 2024), Mistral-7B-Instruct-v0.3 (Mistral AI, 2024), and Gemma-2-9B-it (Team et al., 2024). This selection allows us to assess the applicability of GuardSTR at comparable model scales. For each model, we compare GuardSTR with unsteered inference (Base), Static CAA (Rimsky et al., 2024), and K-CAST (Valentino et al., 2026). Static CAA applies a fixed contrastive steering vector, whereas K-CAST uses a k-nearest-neighbor classifier in activation space to select the sign of the steering coefficient.

Data splits. We use disjoint sets of task inputs for training ($\mathcal{D}_{\text{train}}$), validation ($\mathcal{D}_{\text{val}}$), and testing ($\mathcal{D}_{\text{test}}$). The training split is used to fit the NBF, the validation split to assess candidate guardrails and select hyperparameters, and the test split for the final comparison. All methods use the same fixed partitions. Ground-truth answers are used for offline training, validation, and outcome evaluation; online steering (Section 4) uses only the generated prefix and the learned guardrail.

5.2 NBF Training and Guardrail Validation

Section 3.3 formulates NBF learning over the full state space $\mathcal{S}$. Since possible reasoning prefixes and their latent representations cannot be exhaustively enumerated, we train on recorded trajectories using empirical loss surrogates and assess candidate guardrails on held-out validation trajectories using two empirical metrics.

Task-level training on sampled trajectories. We learn a shared NBF B_ϕ for each task comprising problems $\{x_i\}_{i\in\mathcal{I}}$, avoiding the overhead of training a separate NBF for every problem. The task-level target set is $\mathcal{G} = \bigcup_{i\in\mathcal{I}} \mathcal{G}_i$, where $\mathcal{G}_i$ is the correct-answer target set for x_i. The resulting guardrail is theoretically grounded in the conditional reachability analysis of Proposition 1, reduces computational overhead through shared training, and is empirically supported by our evaluations on held-out problems.

We collect a multiset of unsteered trajectories $\mathcal{R}_{\text{train}}$ from $\mathcal{D}_{\text{train}}$ under a fixed decoding configuration. Following Section 3.1, we extract states and pool observations across problems into three training multisets. The target multiset $\mathcal{S}^{\mathcal{G}}_{\text{train}}$ contains terminal states whose extracted answers are correct for their associated problems. The non-target multiset $\mathcal{S}^{\neg\mathcal{G}}_{\text{train}}$ contains all other states, including prompt and intermediate states and terminal states with incorrect, missing, or invalid answers. The transition multiset $\mathcal{E}^{\neg\mathcal{G}}_{\text{train}}$ contains consecutive state pairs (s_t, s_{t+1}) observed up to termination. We evaluate the corresponding losses from Section 3.3 as sample averages over these multisets and combine them using Eq. (8) to train the shared NBF.

Validation criteria. We use two empirical metrics. *(i) Minimum bound value.* We compute $\min_{\tau\in\mathcal{R}_{\text{val}}} p_{\text{val}}(\tau)$, where $p_{\text{val}}(\tau)$ evaluates Eq. (4) at the initial state of validation trajectory τ. Its threshold is $\rho_{\min} = 0.5\,|\mathcal{S}^{\mathcal{G}}_{\text{train}}|/|\mathcal{R}_{\text{train}}|$, half the unsteered model's training-rollout success rate. *(ii) Empirical constraint-satisfaction rate.* We compute $\widehat{C}_{\text{val}} = |\mathcal{S}_{\text{val}}|^{-1} \sum_{s\in\mathcal{S}_{\text{val}}} c(s)$ over all validation state occurrences. Here, $c(s) = 1$ if the applicable boundary condition in Eqs. (1)–(2) holds and, for nonterminal states, the observed successor satisfies $B_\phi(s') \geq B_\phi(s) + \epsilon$; otherwise, $c(s) = 0$. The acceptance threshold is 0.75.

Together, these criteria provide a computationally tractable screening rule for test-time steering. We use fixed heuristic thresholds: 0.5 scales the required bound value to the base model's training success rate, while 0.75 requires constraint satisfaction on most held-out state occurrences, with some tolerance for sampled violations. An NBF is accepted only if both criteria hold.

5.3 Main Results

Table 1: Success rates on seven BBH/MMLU tasks (%, mean ± STD over five decoding runs).

Base model	Method	Bool.	Date	Dyck	Geo.	Count	Phys.	Math.
Qwen3.5-9B	Base	86.4±2.2	88.4±4.1	26.0±4.5	75.6±3.8	94.8±1.8	82.8±1.4	87.3±1.1
	Static CAA	86.0±2.8	89.2±3.6	25.6±4.8	76.4±4.1	94.0±2.4	83.0±1.8	87.9±1.4
	K-CAST	87.2±2.7	88.0±4.5	27.2±4.1	75.2±3.6	95.2±1.8	83.6±1.6	86.9±1.5
	GuardSTR	91.2±2.3	94.0±3.2	38.0±3.7	79.6±3.6	98.8±1.1	88.9±1.3	91.5±1.6
Llama-3.1-8B-Instruct	Base	62.4±6.5	56.0±4.7	0.8±1.1	35.2±3.6	56.0±7.7	49.0±4.6	56.1±2.9
	Static CAA	63.6±6.8	55.2±5.0	0.8±1.1	36.0±4.0	54.8±8.2	49.8±4.2	55.7±3.3
	K-CAST	62.0±6.2	57.6±4.3	1.6±1.7	34.8±3.6	57.2±7.4	48.5±4.9	57.0±2.7
	GuardSTR	68.8±5.8	60.4±4.3	11.6±3.0	41.6±3.3	60.4±6.8	53.1±4.5	62.5±2.6
Mistral-7B-Instruct-v0.3	Base	63.6±5.5	31.6±5.9	2.4±1.7	16.4±3.0	24.4±7.3	34.7±3.1	35.1±2.0
	Static CAA	62.8±5.9	32.8±5.4	2.0±1.4	17.2±3.3	23.6±7.8	35.4±3.4	34.8±2.4
	K-CAST	64.8±5.2	30.8±6.3	3.2±1.8	16.0±2.8	25.6±6.8	34.2±3.0	36.0±2.1
	GuardSTR	67.6±5.2	38.0±5.1	14.8±3.0	22.0±2.4	28.4±6.5	40.8±2.8	39.4±2.3
Gemma-2-9B-it	Base	84.4±5.5	76.8±3.9	13.6±2.2	48.0±2.0	68.0±3.7	68.2±3.0	64.7±1.4
	Static CAA	85.6±5.2	76.0±4.5	13.2±2.7	48.8±2.7	69.2±4.1	67.5±3.3	65.1±1.8
	K-CAST	83.6±5.9	78.0±3.5	14.8±2.3	47.6±2.6	67.2±3.6	69.1±2.8	64.2±1.6
	GuardSTR	90.0±4.7	81.2±3.6	22.4±3.0	54.8±2.3	72.4±3.3	73.8±2.7	68.3±1.9

Table 1 compares final-answer success rates across the four base models and seven tasks. GuardSTR achieves the highest mean success rate on all 28 model–task pairs, with macro-average gains of 5.91, 5.80, and 5.61 percentage points over Base, Static CAA, and K-CAST. It achieves maximum gains of 12.8 and 11.6 percentage points over the two steering baselines, Static CAA and K-CAST, respectively. The same setting also yields the largest gain over Base, 12.4 percentage points.

The smallest gain over Base is 3.6 percentage points for Gemma-2-9B-it on High School Mathematics, from 64.7% to 68.3%. Against Static CAA, the minimum gain is 3.2 percentage points, attained by Qwen3.5-9B on Geometric Shapes. Against K-CAST, the minimum gain is 2.8 percentage points, attained by Llama-3.1-8B-Instruct on Date Understanding. Thus, the improvements extend beyond the settings with the largest gains and remain positive across all evaluated model–task pairs. It is worth mentioning that Dyck Languages produces the largest gain for every base model against each of the three comparators, highlighting the benefit of GUARDSTR on difficult task.

5.4 ABLATION ON ADAPTIVE STEERING STRENGTH

We examine the contribution of the guardrail-guided strength selection in Section 4.3. The fixed-mean variant replaces the step-dependent strength α_t^ℓ in Eq. (10) with a constant $\bar{\alpha}$, while retaining the same NBF, feature normalization, intervention layers, and state-dependent steering direction. For each model–task pair, $\bar{\alpha}$ is calibrated on the validation split as the mean adaptive strength over all intervention-eligible step decisions.

We compare the fixed-mean variant with GUARDSTR on all four base models and seven tasks. Both variants use the same NBF checkpoint, data partitions, prompts, decoding configuration, and evaluation procedure. We report the mean and standard deviation of final-answer success rate over five decoding runs, with matched decoding seeds between the variants.

Table 2: Ablation of adaptive steering strength across four base models and seven tasks. Values are final-answer success rates (%, mean ± STD over five decoding runs).

Base model	Method	Bool.	Date	Dyck	Geo.	Count	Phys.	Math.
Qwen3.5-9B	Fixed mean	88.8±2.3	91.6±3.6	31.6±4.1	77.2±3.9	96.8±1.8	85.8±1.5	89.4±1.7
	GUARDSTR	**91.2±2.3**	**94.0±3.2**	**38.0±3.7**	**79.6±3.6**	**98.8±1.1**	**88.9±1.3**	**91.5±1.6**
Llama-3.1-8B-Instruct	Fixed mean	65.6±6.2	58.4±4.8	6.4±2.6	38.4±3.8	57.6±7.3	51.0±4.7	59.2±3.0
	GUARDSTR	**68.8±5.8**	**60.4±4.3**	**11.6±3.0**	**41.6±3.3**	**60.4±6.8**	**53.1±4.5**	**62.5±2.6**
Mistral-7B-Instruct-v0.3	Fixed mean	65.6±5.5	35.2±5.6	8.8±2.7	19.2±3.0	26.0±7.1	37.2±3.2	37.4±2.5
	GUARDSTR	**67.6±5.2**	**38.0±5.1**	**14.8±3.0**	**22.0±2.4**	**28.4±6.5**	**40.8±2.8**	**39.4±2.3**
Gemma-2-9B-it	Fixed mean	87.2±5.0	78.8±3.9	17.6±2.6	51.2±2.7	70.0±3.7	70.9±3.2	66.4±2.1
	GUARDSTR	**90.0±4.7**	**81.2±3.6**	**22.4±3.0**	**54.8±2.3**	**72.4±3.3**	**73.8±2.7**	**68.3±1.9**

As shown in Table 2, the reported means favor adaptive steering over the fixed-mean variant on all 28 model–task pairs, with a macro-average gain of 3.04 percentage points when each pair is weighted equally. The largest gain is 6.4 percentage points for Qwen3.5-9B on Dyck Languages, increasing success from 31.6% to 38.0%.

Dyck Languages shows the largest benefit from adaptive strength for all four models: 6.4, 5.2, 6.0, and 4.8 percentage points for Qwen3.5-9B, Llama-3.1-8B-Instruct, Mistral-7B-Instruct-v0.3, and Gemma-2-9B-it, respectively. Because the fixed-mean variant retains the same NBF and state-dependent direction rule, these results support the additional benefit of adapting steering strength to the evolving reasoning state.

6 CONCLUSION

We presented GUARDSTR, a framework that uses neural reasoning barrier functions as guardrails for adaptive CoT steering. By formulating reasoning as a finite-horizon stochastic reachability problem, we connect local boundary and transition conditions to a lower bound on correct-answer probability. Learned from outcome-labeled trajectories without intermediate-step annotations, the barrier provides transition-level feedback for adapting intervention strength and direction. Steering leaves the base model's parameters fixed and requires no ground-truth answers at inference time. Across four LLMs and seven reasoning tasks, GUARDSTR improves final-answer accuracy over unsteered inference on all 28 model–task pairs, with a macro-average gain of 5.91 percentage points. A limitation is that our empirical validation of the barrier conditions is restricted to sampled trajectories. Future work will investigate stronger verification of these conditions and extend guardrail steering to broader reasoning settings, including agentic workflows.

7 AI USE STATEMENT

Large language models (LLMs) were used solely to improve the manuscript's language, clarity, and readability. They played no role in research ideation, methodology, experimental design, or data analysis; all scientific content was developed by the authors. The evaluated language models also generated the chain-of-thought trajectories used for NBF training. The authors take full responsibility for the theoretical claims, methodological decisions, implementation, analysis, and final manuscript, including all content developed with AI assistance.

8 ETHICS STATEMENT

We are committed to following the ICLR Code of Ethics and maintaining transparency and integrity throughout the research process. This study uses publicly available BBH and MMLU benchmarks (Suzgun et al., 2023; Hendrycks et al., 2021) and existing language models, without conducting a new human-subject study or animal experimentation. The use and redistribution of these resources remain subject to their respective licenses and usage conditions. We acknowledge that the evaluated models may exhibit biases or produce incorrect outputs, and that improvements on the selected reasoning tasks do not establish general safety or reliability. Applications beyond the evaluated settings require further assessment of potential risks and unintended effects.

9 REPRODUCIBILITY STATEMENT

To support independent reproduction and verification, we document the experimental environment in Appendix B.1, the dataset partitions in Appendix B.2, and the prompt templates and answer-format instructions in Appendix B.4. Appendices B.5 and B.6 describe the experiments on steering strength and intervention-layer selection, respectively. The BBH and MMLU benchmarks are publicly available. We will also provide anonymized code, experiment configurations, dataset-split identifiers, and run-level evaluation results as supplementary material with the submission, enabling researchers to inspect the experimental procedure and recompute the reported statistics.

REFERENCES

Charles Dawson, Zengyi Qin, Sicun Gao, and Chuchu Fan. Safe nonlinear control using robust neural lyapunov-barrier functions. In Aleksandra Faust, David Hsu, and Gerhard Neumann (eds.), *Conference on Robot Learning, 8-11 November 2021, London, UK*, volume 164 of *Proceedings of Machine Learning Research*, pp. 1724–1735. PMLR, 2021. URL https://proceedings.mlr.press/v164/dawson22a.html.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Kentaro Kazama, Daiki Shirafuji, and Tatsuhiko Saito. Geosteer: Faithful chain-of-thought steering via latent manifold gradients. *CoRR*, abs/2601.10229, 2026. doi: 10.48550/ARXIV.2601.10229. URL https://doi.org/10.48550/arXiv.2601.10229.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/8bb0d291acd4acf06ef112099c16f326-Abstract-Conference.html.

Evgenii Kortukov, Piotr Komorowski, Florian Klein, Paula Engl, Gabriele Sarti, Seong Joon Oh, Sebastian Lapuschkin, and Wojciech Samek. Predicting future behaviors in reasoning models enables better steering. *CoRR*, abs/2606.11172, 2026. doi: 10.48550/ARXIV.2606.11172. URL https://doi.org/10.48550/arXiv.2606.11172.

Bruce W. Lee, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Erik Miehling, Pierre L. Dognin, Manish Nagireddy, and Amit Dhurandhar. Programming refusal with conditional activation steering. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=Oi47wc10sm.

Mistral AI. Mistral-7B-Instruct-v0.3. Hugging Face model card, 2024. URL https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3. Accessed: 2026-09-26.

Tuc Nguyen and Thai Le. Atlas: Verifier-guided adaptive latent activation steering for efficient llm reasoning. *arXiv preprint arXiv:2601.03093*, 2026.

Stephen Prajna, Ali Jadbabaie, and George J. Pappas. A framework for worst-case and stochastic safety verification using barrier certificates. *IEEE Trans. Autom. Control.*, 52(8):1415–1428, 2007. doi: 10.1109/TAC.2007.902736. URL https://doi.org/10.1109/TAC.2007.902736.

Zengyi Qin, Kaiqing Zhang, Yuxiao Chen, Jingkai Chen, and Chuchu Fan. Learning safe multi-agent control with decentralized neural barrier certificates. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=P6_q1BRxY8Q.

Qwen Team. Qwen3.5: Towards native multimodal agents. Qwen blog, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. Steering llama 2 via contrastive activation addition. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 15504–15522. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.828. URL https://doi.org/10.18653/v1/2024.acl-long.828.

Ali Salamati and Majid Zamani. Data-driven safety verification of stochastic systems via barrier certificates: A wait-and-judge approach. In Roya Firoozi, Negar Mehr, Esen Yel, Rika Antonova, Jeannette Bohg, Mac Schwager, and Mykel J. Kochenderfer (eds.), *Learning for Dynamics and Control Conference, L4DC 2022, 23-24 June 2022, Stanford University, Stanford, CA, USA*, volume 168 of *Proceedings of Machine Learning Research*, pp. 441–452. PMLR, 2022. URL https://proceedings.mlr.press/v168/salamati22a.html.

John Scoville, Shengzhuang Chen, Yejin Bang, Stefan Winzeck, and Jonathan Richard Schwarz. Cognitive demand steering for adaptive meta-reasoning in large language models. *CoRR*, abs/2608.01319, 2026. doi: 10.48550/ARXIV.2608.01319. URL https://doi.org/10.48550/arXiv.2608.01319.

Yuan Sui, Yufei He, Tri Cao, Sophia Simeng Han, Yulin Chen, and Bryan Hooi. Meta-reasoner: Dynamic guidance for optimized inference-time reasoning in large language models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics, ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 13268–13286. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.FINDINGS-ACL.649. URL https://doi.org/10.18653/v1/2026.findings-acl.649.

Lihao Sun, Hang Dong, Bo Qiao, Qingwei Lin, Dongmei Zhang, and Saravan Rajmohan. LLM reasoning as trajectories: Step-specific representation geometry and correctness signals. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL*

2026, San Diego, California, United States, July 2-7, 2026, pp. 26872–26887. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.ACL-LONG.1237. URL https://doi.org/10.18653/v1/2026.acl-long.1237.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, volume ACL 2023 of *Findings of ACL*, pp. 13003–13051. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-ACL.824. URL https://doi.org/10.18653/v1/2023.findings-acl.824.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Thanh Q. Tran, Arun Verma, Kiwan Wong, Bryan Kian Hsiang Low, Daniela Rus, and Wei Xiao. Barriersteer: LLM safety via learning barrier steering. *CoRR*, abs/2602.20102, 2026. doi: 10.48550/ARXIV.2602.20102. URL https://doi.org/10.48550/arXiv.2602.20102.

Alexander Matt Turner, Lisa Thiergart, David Udell, Gavin Leech, Ulisse Mini, and Monte MacDiarmid. Activation addition: Steering language models without optimization. *CoRR*, abs/2308.10248, 2023. doi: 10.48550/ARXIV.2308.10248. URL https://doi.org/10.48550/arXiv.2308.10248.

Marco Valentino, Geonhee Kim, Dhairya Dalal, Zhixue Zhao, and André Freitas. Mitigating content effects on reasoning in language models through fine-grained activation steering. In Sven Koenig, Chad Jenkins, and Matthew E. Taylor (eds.), *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pp. 33314–33322. AAAI Press, 2026. doi: 10.1609/AAAI.V40I39.40617. URL https://doi.org/10.1609/aaai.v40i39.40617.

Tianlong Wang, Yuhang Wang, Weibin Liao, Xin Gao, Xinyu Ma, Yang Lin, Yasha Wang, and Liantao Ma. Search for truth from reasoning: A dynamic representation editing framework for steering LLM trajectories. *CoRR*, abs/2606.28589, 2026. doi: 10.48550/ARXIV.2606.28589. URL https://doi.org/10.48550/arXiv.2606.28589.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

Bai Xue. Finite-time safety and reach-avoid verification of stochastic discrete-time systems. *Inf. Comput.*, 307:105368, 2025. doi: 10.1016/J.IC.2025.105368. URL https://doi.org/10.1016/j.ic.2025.105368.

Dapeng Zhi, Peixin Wang, Si Liu, C.-H. Luke Ong, and Min Zhang. Unifying qualitative and quantitative safety verification of dnn-controlled systems. In Arie Gurfinkel and Vijay Ganesh (eds.), *Computer Aided Verification - 36th International Conference, CAV 2024, Montreal, QC, Canada, July 24-27, 2024, Proceedings, Part II*, volume 14682 of *Lecture Notes in Computer Science*, pp. 401–426. Springer, 2024. doi: 10.1007/978-3-031-65630-9_20. URL https://doi.org/10.1007/978-3-031-65630-9_20.

Andy Zou, Long Phan, Sarah Li Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt

Fredrikson, J. Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to AI transparency. *CoRR*, abs/2310.01405, 2023. doi: 10.48550/ARXIV.2310.01405. URL https://doi.org/10.48550/arXiv.2310.01405.

A Derivation of the Finite-Horizon Reachability Bound

This appendix reconstructs the proof structure of Theorem 6 in UniQQ (Zhi et al., 2024), using our reasoning-system notation.

Definition 1 (Reasoning Barrier Functions). *A measurable function $B_\phi : \mathcal{S} \to \mathbb{R}_{\geq 0}$ is a reasoning barrier function for the reasoning stochastic process $\mathcal{M}$ if there exist constants $\beta \in (0, 1)$, $\beta < \alpha < 1 + \beta$, and $\epsilon \geq 0$ such that*

$$B_\phi(s) \leq \beta, \qquad \forall s \in \mathcal{S} \setminus \mathcal{G}_x, \tag{1}$$

$$\alpha \leq B_\phi(s) \leq 1 + \beta, \qquad \forall s \in \mathcal{G}_x, \tag{2}$$

$$\mathbb{E}[B_\phi(s_{t+1}) \mid s_t = s] \geq B_\phi(s) + \epsilon, \qquad \forall s \in \mathcal{S} \setminus \mathcal{G}_x. \tag{3}$$

Proposition 1 (Finite-Horizon Correct-Answer Reachability). *Suppose B_ϕ satisfies Definition 1. Under the correspondence between target membership and successful termination specified above, the success probability satisfies that for any trajectory starting from s_0*

$$p_{\text{succ}} = \Pr(\exists\, t \in \{1, \ldots, T_{\max}\} : s_t \in \mathcal{G}_x) \geq \max\left\{0,\ B_\phi(s_0) + \frac{1}{2}\epsilon T_{\max} - \beta\right\}. \tag{4}$$

Fix an initial state $s_0 \in \mathcal{S}_0$ and a maximum CoT horizon $T_{Max} \in \mathbb{N}_{>0}$. Throughout, probabilities and expectations are conditioned on s_0.

Define the first time at which the process reaches the correct-answer target set by

$$\kappa = \inf\{t \geq 0 : s_t \in \mathcal{G}_x\},$$

with $\inf \emptyset = \infty$, and define

$$\widetilde{s}_t = s_{\min(t,\kappa)}.$$

Thus, the stopped process retains its state after reaching $\mathcal{G}_x$. The finite-horizon reachability probability satisfies

$$p_T := \mathbb{P}_{s_0}(\exists t \leq T_{Max} : s_t \in \mathcal{G}_x) = \mathbb{P}_{s_0}(\widetilde{s}_T \in \mathcal{G}_x). \tag{13}$$

Let $p_t = \mathbb{P}_{s_0}(\widetilde{s}_t \in \mathcal{G}_x)$. Since the stopped process remains in the target set after entry,

$$p_0 \leq p_1 \leq \cdots \leq p_{T_{Max}}.$$

Consequently,

$$p_{T_{Max}} \geq \frac{1}{T_{Max} + 1} \sum_{t=0}^{T_{Max}} p_t. \tag{14}$$

The upper bounds inside and outside $\mathcal{G}_x$ imply the pointwise inequality

$$B_\phi(\widetilde{s}_t) \leq \beta + \mathbb{I}\{\widetilde{s}_t \in \mathcal{G}_x\}.$$

Taking expectations and rearranging gives

$$p_t \geq \mathbb{E}_{s_0}[B_\phi(\widetilde{s}_t)] - \beta. \tag{15}$$

Let $\widetilde{\mathcal{F}}_t = \sigma(\widetilde{s}_0, \ldots, \widetilde{s}_t)$.

$$\mathbb{E}_{s_0}[B_\phi(\widetilde{s}_{t+1}) \mid \widetilde{\mathcal{F}}_t] - B_\phi(\widetilde{s}_t) \geq \epsilon, \qquad 0 \leq t < T_{Max}. \tag{16}$$

Retaining this assertion, the law of total expectation gives

$$\mathbb{E}_{s_0}[B_\phi(\widetilde{s}_{t+1})] \geq \mathbb{E}_{s_0}[B_\phi(\widetilde{s}_t)] + \epsilon.$$

Iterating from $\widetilde{s}_0 = s_0$ yields

$$\mathbb{E}_{s_0}[B_\phi(\widetilde{s}_t)] \geq B_\phi(s_0) + \epsilon t. \tag{17}$$

Substituting Eqs. (15) and (17) into Eq. (14), the argument becomes

$$\begin{aligned} p_{T_{Max}} &\geq \frac{1}{T_{Max}+1} \sum_{t=0}^{T_{Max}} p_t \\ &\geq \frac{1}{T_{Max}+1} \sum_{t=0}^{T_{Max}} (\mathbb{E}_{s_0}[B_\phi(\widetilde{s}_t)] - \beta) \\ &\geq \frac{1}{T_{Max}+1} \sum_{t=0}^{T_{Max}} (B_\phi(s_0) + \epsilon t - \beta) \\ &= B_\phi(s_0) - \beta + \frac{\epsilon}{T_{Max}+1} \frac{T_{Max}(T_{Max}+1)}{2} \\ &= B_\phi(s_0) + \frac{1}{2}\epsilon T_{Max} - \beta. \end{aligned} \tag{18}$$

Combining this derived inequality with $p_{T_{Max}} \geq 0$ gives the requested form of the conclusion:

$$\boxed{\mathbb{P}_{s_0}(\exists t \leq T_{Max} : s_t \in \mathcal{G}_x) \geq \max\left\{0,\ B_\phi(s_0) + \frac{1}{2}\epsilon T_{Max} - \beta\right\}.} \tag{19}$$

B Experiment Details

B.1 Models and execution environment

The model suite comprises Qwen3.5-9B, Llama-3.1-8B-Instruct, Mistral-7B-Instruct-v0.3, and Gemma-2-9b-it. The execution configuration uses one NVIDIA RTX PRO 6000 Blackwell Workstation Edition GPU with 97,887 MiB of device memory. Unsteered Base generation and offline trajectory collection use vLLM 0.25.0 with tensor parallelism 1, GPU-memory utilization 0.85, BF16 model weights, a maximum sequence length of 8,576 tokens, and prefix caching disabled. Hidden-state extraction and activation-controlled decoding use Transformers 5.12.0 in a separate controller environment with PyTorch 2.12.0. Controlled decoding maintains a continuing key–value cache; model parameters remain frozen. Barrier evaluation, hidden-state gradient computation, and the Transformers sampling logits use FP32.

Qwen3.5-9B is loaded through its text-only causal language-model component. For Gemma 2, the runtime `max_position_embeddings` setting is increased from 8,192 to 8,576 to accommodate the common sequence budget. This changes the runtime configuration without updating pretrained weights.

All methods use temperature 0.8, top-p 0.95, and a maximum of 8,192 generated tokens. Prompt and completion lengths must jointly fit within the 8,576-token context budget. The five decoding runs use seeds $\{42, 43, 44, 45, 46\}$ with fixed data partitions and fitted method parameters. For run seed s, the request seed for zero-based test-row index i is $s + 1{,}000{,}003i$. Reported variability is the sample standard deviation across the five run-level success rates, with denominator $5 - 1$; it does not include retraining the NBF with different initialization seeds. The same sampling settings and seed schedule are used across methods, while Base and controlled decoding retain their respective execution backends.

B.2 Datasets and splits

Each selected BBH task contains 250 examples. For Boolean Expressions, Dyck Languages, Geometric Shapes, and Object Counting, row indices are shuffled using a local Python random generator with seed 42. The first 150 rows form the training set, the next 50 form the validation set, and the remaining 50 form the test set.

Date Understanding contains questions derived from shared calendar scenarios. The scenario key is the portion of a question preceding its final `What is the date` clause. A grouped, seeded subset-sum allocation assigns each scenario to exactly one partition while preserving the $150/50/50$

Table 3: Dataset partitions shared by all compared methods. "Row" denotes a seeded row split; "grouped" keeps related Date scenarios together; "official" retains the native MMLU dev, validation, and test partitions, with dev used for training. All BBH split seeds are 42. Eval. N is the number of test examples in each decoding run.

Benchmark	Task	Split rule	Train	Validation	Test	Eval. N
BBH	Boolean Expressions	row	150	50	50	50
BBH	Date Understanding	grouped	150	50	50	50
BBH	Dyck Languages	row	150	50	50	50
BBH	Geometric Shapes	row	150	50	50	50
BBH	Object Counting	row	150	50	50	50
MMLU	High School Physics	official	5	17	151	151
MMLU	High School Mathematics	official	5	29	270	270

row counts. The training, validation, and test partitions contain 25, 8, and 8 disjoint scenario groups, respectively.

For each MMLU subject, we retain the official dev, validation, and test partitions without pooling or repartitioning. The official dev partition serves as the training set, while the official validation and test partitions serve as the validation and test sets, respectively. High School Physics contains 173 examples, with training/validation/test counts of $5/17/151$; High School Mathematics contains 304 examples, with counts of $5/29/270$. All examples are retained exactly once, with their original source-partition and row identifiers preserved.

All methods share these partitions. Training examples are used for method fitting; validation examples are used for NBF acceptance, baseline coefficient selection, and fixed-mean calibration. Final-answer evaluation uses the entire resulting test partition. Training and validation examples are not used as in-context demonstrations. Table 3 summarizes the partitions and per-run evaluation populations.

B.3 Evaluation metric

Our primary test metric is final-answer success rate, corresponding to the empirical frequency of correct-answer reachability within the generation budget defined in Section 3.1. For the $N_{\text{test}} = |\mathcal{D}_{\text{test}}|$ test examples, a run is successful when its extracted final answer matches the ground-truth answer after benchmark-specific normalization. For BBH, we use the official task-specific answer normalization and exact-match scorer. For MMLU, we extract the selected option label (A–D) and compare it with the gold label. Missing or invalid answers, including budget exhaustion without an accepted correct answer, are counted as failures. We report the mean and standard deviation of success rate over five independent decoding runs.

B.4 Prompt templates

All compared methods use the same zero-shot raw-text prompt for a given task. The prompt is passed directly to the tokenizer without a chat template, system message, or in-context demonstration. Tokenizer-defined special-token insertion is enabled. The exact outer template is

```
Please reason step by step, and end with
{task_final_format}.
User: {problem}
Assistant: <think>
```

For Date Understanding, Base, Static CAA, K-CAST, GUARDSTR, and the fixed-mean ablation instantiate this template as follows:

```
Please reason step by step, and end with the exact option
format 'Final answer: (X)'.
User: {problem}
Assistant: <think>
```

Table 4: Exact final-answer instruction inserted into the common prompt.

Task	task_final_format
Boolean Expressions	the exact format 'Final answer: True' or 'Final answer: False'
Date Understanding	the exact option format 'Final answer: (X)'
Dyck Languages	'Final answer:' followed only by the missing closing-bracket sequence
Geometric Shapes	the exact option format 'Final answer: (X)'
Object Counting	the exact integer format 'Final answer: N'
High School Physics	the exact option format 'Final answer: (X)'
High School Mathematics	the exact option format 'Final answer: (X)'

Table 4 lists the exact task-specific substitutions. The period following each substitution belongs to the outer template.

For MMLU, the `problem` field contains the question and its four answer choices, without additional prose:

```
{question}
Options:
(A) {choice_0}
(B) {choice_1}
(C) {choice_2}
(D) {choice_3}
```

No stop string is attached to `Final answer:`. Decoding stops at the tokenizer EOS token or the generation-length limit, and answer extraction uses the complete generated text.

Reasoning-step boundaries are identified by explicit line-leading `Step i:` markers. The first marker starts the first segment; completion of a subsequent marker closes the preceding segment, and termination closes the final segment. The prompt does not add a separate instruction requiring these markers. A response without such markers forms one terminal segment and has no eligible intermediate steering boundary. The reasoning horizon is capped at 75 segments; exceeding this horizon raises an explicit error rather than silently truncating the trajectory.

B.5 Effect of Steering Strength

The steering-strength sweep uses Qwen3.5-9B on Boolean Expressions and Date Understanding. The global coefficient in Eq. (10) takes values $c_\alpha \in \{0, 0.25, 0.5, 1.0, 1.5, 2.0\}$. For the nonzero coefficients, the steering layer is fixed at $\ell = 1$, and all conditions reuse the same accepted, task-specific NBF checkpoint and feature normalization. Prompts, data partitions, decoding settings, and run seeds are held fixed.

At $c_\alpha = 0$, the activation perturbation vanishes. The zero-strength reference in the experiment registry reuses the unsteered vLLM Base runs; nonzero-strength conditions use the Transformers steering implementation. The default coefficient is configured as $c_\alpha = 1.0$.

This sweep isolates the magnitude of the activation correction within the controlled-decoding conditions. The correction follows the normalized positive gradient of the success-oriented barrier. Changing c_α therefore changes the perturbation magnitude while preserving the causal credit rule and correction direction. Figure 3 provides the corresponding comparison across the prescribed coefficients.

B.6 Effect of the Steering Layer

The layer sweep uses Qwen3.5-9B on the same two tasks. With one-based indexing of its 32 decoder blocks, the candidate layers are $\ell \in \{1, 8, 16, 24, 32\}$. Steering strength is fixed at $c_\alpha = 1.0$, while data partitions, prompts, training hyperparameters, and decoding settings are shared across conditions.

Each candidate layer has its own NBF, fitted on training representations from that layer and assessed on the corresponding validation trajectories. A checkpoint must satisfy both validation conditions before it is used for test-time steering. For intermediate layers, the representation is the last-token

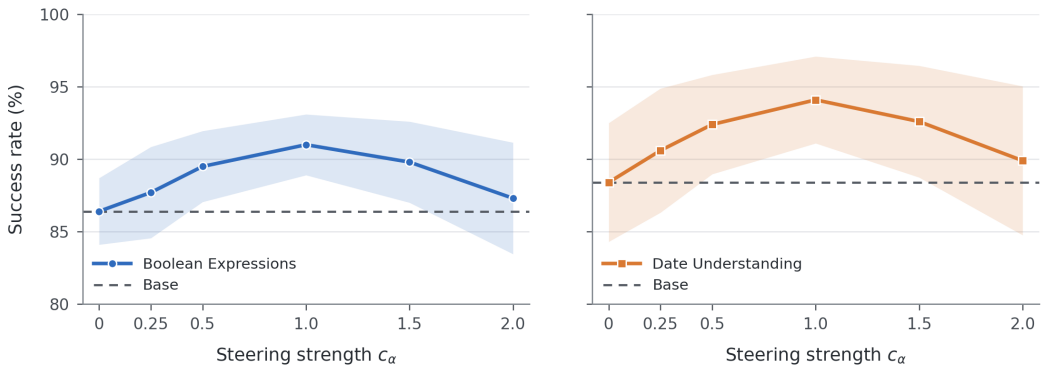

Figure 3: Steering-strength sensitivity of Qwen3.5-9B on Boolean Expressions and Date Understanding. Nonzero-strength conditions use layer $\ell = 1$ and a shared NBF checkpoint within each task. Lines summarize mean final-answer success over five decoding runs, shaded regions indicate one sample standard deviation, and dashed horizontal lines denote the corresponding unsteered Base reference.

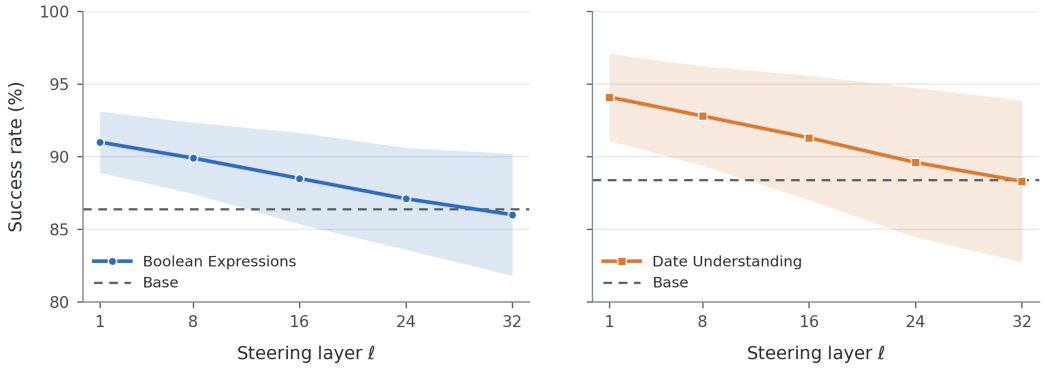

Figure 4: Steering-layer sensitivity of Qwen3.5-9B on Boolean Expressions and Date Understanding at $c_\alpha = 1.0$. Layer indices are one-based; each layer uses its own fitted and validated NBF. Lines summarize mean final-answer success over five decoding runs, shaded regions indicate one sample standard deviation, and dashed horizontal lines denote the corresponding unsteered Base reference.

output of the selected block. For $\ell = 32$, the implementation uses the last-token representation after the decoder's final normalization. The default intervention layer is configured as $\ell = 1$.

Figure 4 compares intervention locations under the same strength and evaluation protocol. The sweep covers an early decoder block, intermediate blocks, and the final normalized representation, allowing the effect of intervention depth to be examined together with its associated layer-specific barrier.

B.7 Signed score computation

At each reasoning step t, $\widehat{A}_t^-$ and $\widehat{A}_t^+$ quantify, respectively, the normalized degree of violation of the NBF constraints and the normalized margin by which these constraints are satisfied. Their magnitudes characterize the strength of the corresponding constraint-based signals and serve as weights reflecting the influence of the current step on the final reasoning outcome. To summarize their relative dominance while retaining both the direction and strength of the dominant signal, we

define the signed guardrail score

$$A_t = \begin{cases} -\widehat{A}_t^-, & \widehat{A}_t^- > \widehat{A}_t^+, \\ +\widehat{A}_t^+, & \widehat{A}_t^+ \geq \widehat{A}_t^-. \end{cases} \tag{20}$$

The sign of A_t identifies whether the violation or surplus signal dominates, with ties assigned to the surplus branch. Its magnitude, $|A_t| = \max\{\widehat{A}_t^-, \widehat{A}_t^+\}$, determines the weight of the dominant signal: a larger absolute score indicates a stronger influence of the current step on the final reasoning outcome.

B.8 A Running Example of GuardSTR

Figure 5 illustrates Qwen3.5-9B on a Date Understanding example, showing three selected negatively scored steps before steering and three positively scored steps after steering. Following Eq. (20), the sign and magnitude of A_t indicate the direction and strength of the step's estimated influence on final-answer correctness.

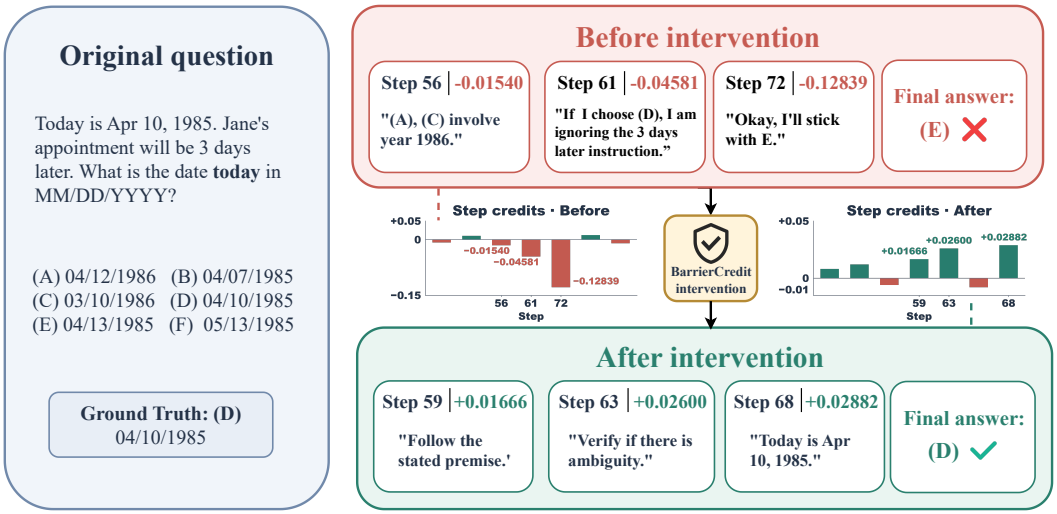

Figure 5: A date-reasoning example of GuardSTR.

The question asks for today's date, April 10, 1985, while mentioning an appointment three days later. Before steering, Step 56 excludes options with the wrong year, but Step 61 mistakes the appointment information for the requested date. Step 72 then commits to the incorrect answer (E), receiving the most negative score among the highlighted steps. After steering, Steps 59 and 63 revisit the premise and check its interpretation. Step 68 explicitly recovers today's date and receives the largest displayed positive score, followed by the correct answer (D). These scores highlight estimated adverse and supportive influences along the trajectory, providing a contextual interpretation of the guardrail's assessment.