Reasoning Models Reason Inefficiently

Abstract

Large language models (LLMs) produce long, structured reasoning traces that can inflate latency and cost. Our results suggest that while backtracking can help models arrive to the correct answer, they are not a faithful picture of the minimal computation required to solve a task—they can be compressed or restructured. In this paper, we show how to build more efficient and interpretable reasoning processes by identifying and targeting internal directions associated with inefficiency. For reproducibility, our code and evaluation scripts are publicly accessible at `<my-github-link-here>`.

1 Introduction

We demonstrate steering and projection interventions that move models to strictly better points on the mean tokens, accuracy plane, evidence of avoidable computation. We conducted experiments on the OpenR1-Math-220k dataset [Open R1 Project, 2025], a large-scale collection of 220,000 verified mathematical problems. Across 2,000 samples from the `open-r1/OpenR1-Math-220k` dataset, our interventions reduced both output length and the number of tokens until answer without hurting exact-match accuracy.

We find that steering away from backtracking reduces the average output length and the number of tokens until answer, without reducing exact-match accuracy. Prior work has suggested that backtracking can serve as a useful proxy for self-correction [Venhoff et al., 2025], but also noted inefficiencies due to overthinking in reasoning traces [Anonymous, 2025].

This demonstrates that reasoning models can solve problems more succinctly than their default reasoning traces reveal, highlighting inefficiencies in current reasoning modes.

2 Related Work

Backtracking as intentional behavior: Chain-of-Thought (CoT) prompting generates step-by-step rationales and improves accuracy in multistep reasoning tasks [Wei et al., 2022]. Self-consistency further increases CoT through the sampling of multiple traces and the aggregate of answers [Wang et al., 2023]. Recent reasoning models explicitly optimize for long structured thoughts, making length/accuracy trade-offs central to the evaluation [DeepSeek-AI, 2025].

Longer inference can sometimes reduce accuracy and induce rumination, suggesting inefficiency regimes in "reasoning mode" [Anonymous, 2024]. Our work complements these by showing activation-space interventions that move models to better points on the tokens–accuracy frontier. Because we do not rely on inference-time interventions but instead directly modify the model itself, we open-source our more efficient model to facilitate further research and reproducibility[1].

Steering vectors and representation editing: A growing body of work explores steering vectors, task arithmetic, and representation editing to control LLM behavior [Ilharco et al., 2022, Rajamanoharan et al., 2025]. Most existing studies focus on small-scale tasks, synthetic probes, or qualitative

[1] Model release available at: `<my-hf-link-here>`

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Efficient Reasoning.

demonstrations, flipping sentiment or refusals for example, Few evaluate whether such methods yield measurable benefits on real reasoning workloads. By quantifying efficiency improvements on a large math reasoning dataset (`open-r1/OpenR1-Math-220k`), our work shows that steering interventions can reduce computation while preserving accuracy, providing the kind of dataset-level evidence needed for deployment in practical reasoning pipelines.

3 Methodology

3.1 Steering Vectors

To obtain candidate steering vectors, we collected a balanced dataset of $10{,}000$ reasoning traces from the `LLaMA-8B` model. Each example was processed with a fixed "step-by-step" suffix to elicit chain-of-thought reasoning. For every layer ℓ in the model, we extracted the hidden activations associated with two conditions of interest: (i) traces exhibiting overt backtracking cues (e.g., "*Wait, actually...*"), and (ii) traces without such cues.

Formally, let $\{h_{i,\ell}^{\text{back}}\}_{i=1}^{N}$ denote activations from condition (i) at layer ℓ, and $\{h_{j,\ell}^{\text{non}}\}_{j=1}^{M}$ those from condition (ii). We compute mean activations for each condition:

$$\mu_\ell^{\text{back}} = \tfrac{1}{N}\sum_{i=1}^{N} h_{i,\ell}^{\text{back}}, \qquad \mu_\ell^{\text{non}} = \tfrac{1}{M}\sum_{j=1}^{M} h_{j,\ell}^{\text{non}}.$$

The steering vector at layer ℓ is defined as the normalized difference:

$$u_\ell = \frac{\mu_\ell^{\text{back}} - \mu_\ell^{\text{non}}}{\|\mu_\ell^{\text{back}} - \mu_\ell^{\text{non}}\|_2}.$$

Each u_ℓ thus represents the axis in hidden space that maximally separates backtracking from non-backtracking activations (see Appendix Figure 3 for PCA visualizations across layers). These vectors form the basis for the intervention operators described next.

3.2 Transformation Operators

Given a weight matrix W and a steering vector u, we define a general parametric operator:

$$W' = (I + \beta\, uu^\top)W, \quad \beta \in \mathbb{R}.$$

This family of linear transforms modifies the component of W aligned with u:

- **Suppression (Projection Removal / Orthogonalization).** Setting $\beta = -1$ removes the u-aligned component, ensuring $W'u = 0$, i.e. the weights no longer express the backtracking direction.
- **Amplification (Directional Scaling).** For arbitrary $\beta = \alpha$, this scales the u-aligned component by $(1 + \alpha)$. Positive α strengthens backtracking alignment, while negative α weakens it.

These represent complementary ways of altering the model's use of the steering direction u. *Suppression* removes the contribution of u entirely, testing whether performance depends on the presence of this direction.*Amplification* continuously strengthens or weakens the u component, letting us explore graded causal effects rather than all-or-nothing edits. We test whether the steering vector encodes a causal feature of the model's reasoning process. Applied at different layers, these edits allow us to examine where in the computation its influence is most critical.

3.3 Metrics

To quantify the effects, we evaluate interventions using the following metrics:

(1) Exact-match accuracy: Let y be the gold answer and $\hat{y}$ the model's extracted final answer. An example is correct iff $\hat{y} = y$ after canonicalization (lowercasing, whitespace trim, and task-specific normalization, e.g., stripping `\boxed{}` or units where applicable).

(2) Output length: We report the mean number of output tokens $\mathbb{E}[\text{len}_{\text{tok}}(\hat{x})]$, where tokenization is computed with the model's native tokenizer. When relevant, we also report *tokens-until-answer*: the number of generated tokens up to and including the first occurrence of the extracted correct answer. To capture the trade-off between correctness and efficiency, we analyze the empirical Pareto frontier of accuracy versus output length, highlighting methods that achieve higher accuracy with shorter generations.

(3) Backtracking rate at sentence starts: We measure the fraction of generated sentences that begin with backtracking cues such as *"wait," "actually," "let me redo," "on second thought"* (full list in Appendix Table 2). This provides a coarse proxy for how often models explicitly revise or undo their reasoning mid-trace.

4 Results

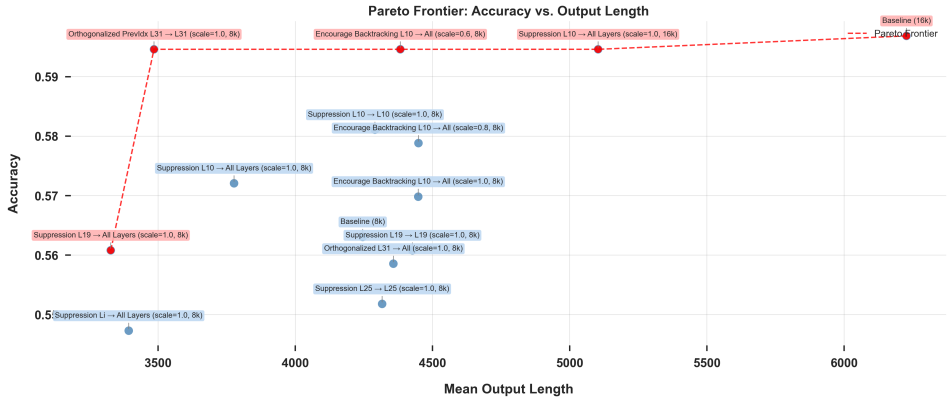

Figure 1: Pareto frontier of accuracy vs. mean output tokens. Points on the frontier maximize accuracy while minimizing length. Amplification (α=0.6, 8k) and Suppression (L31→L31, 8k) are Pareto-optimal

4.1 Accuracy Across Steering Methods

Table 1 compares accuracy and efficiency across interventions. The strongest accuracy is *Baseline (16k)* at 59.7%. Several edits match this while using fewer tokens: *Suppression L10 → All (16k)* attains 59.5% (only -0.2 pts vs. baseline) with shorter outputs (5103 vs. 6227 mean tokens); *Amplification (α=0.6) L10 → All (8k)* also reaches 59.5% at 4383 mean tokens; and *Suppression PrevIdx L31 → L31 (8k)* matches 59.5% with just 3485 tokens on average. These results show that targeted steering can recover baseline-level accuracy without extended context or long generations.

4.2 Backtracking Cue Frequency

Transformations reducing sentence-initial backtracking cues in Figure 2 are the same edits that shorten outputs (Suppression L31→L31: 3485 tokens vs. Baseline 6227) while maintaining accuracy (59.5%) in Table 1. Amplification keeps accuracy high but leaves more backtracking cues than Suppression, consistent with its longer generations than the best Suppression runs.

4.3 Pareto Frontier: Accuracy vs. Length

The frontier highlights two efficient regimes: (i) Lightweight Amplification (L10, α=0.6, 8k) and (ii) Layer-specific Suppression (L31→L31, 8k). Both match top accuracy (59.5%) at substantially fewer tokens than 16k baselines (4383 and 3485 vs. 6227).

Overall, these results show that steering interventions can rival or even match extended context baselines while using significantly fewer tokens. The most effective strategies involve either (1) lightweight backtracking ($\alpha = 0.6$) or (2) layer-specific suppression (L31), both of which lie on the Pareto frontier.

Edit	Applied layers	Vector source	MaxTok	Correct (%)	Incorrect (%)	Empty (%)	Avg Len	Tokens to Ans
Baseline								
Baseline	–	–	16k	**59.7**	27.5	12.8	6226.9	1251.3
Baseline	–	–	8k	56.3	33.6	10.1	4245.7	883.7
Supress Backtracking ($W' = (I - uu^\top)W$)								
Supression	All	L10	16k	**59.5**	31.8	8.8	5103.5	975.1
Supression	L31 → L31	L31	8k	59.5	31.3	9.2	3485.2	889.6
Supression	Single L10	L10	8k	58.1	34.9	7.0	4290.2	894.4
Supression	All	L10	8k	57.2	36.7	6.1	3776.9	830.7
Supression	Single L19	L19	8k	56.1	34.9	9.0	4426.9	804.3
Supression	All	L19	8k	56.1	34.0	9.9	3327.6	793.0
Supression	All	L31	8k	55.9	33.3	10.8	4357.5	828.4
Supression	Single L25	L25	8k	55.2	35.6	9.2	4316.5	884.1
Supression	All	Per-layer (own)	8k	54.7	36.7	8.6	3393.1	768.1
Amplify Backtracking ($W' = W + \alpha\, uu^\top W$)								
Amplification ($\alpha = 0.6$)	All L10	L10	8k	**59.5**	31.3	9.2	4382.8	1023.8
Amplification ($\alpha = 0.8$)	All L10	L10	8k	57.9	32.7	9.5	4448.8	977.3
Amplification ($\alpha = 1.0$)	All L10	L10	8k	57.0	35.6	7.4	4448.1	976.1

Table 1: Accuracy and efficiency across steering interventions. While the *Baseline (16k)* achieves the highest raw accuracy (59.7%), several targeted edits match this performance while requiring fewer tokens. In particular, *Suppression L10→All (16k)*, *Amplification* (α=0.6) *L10→All (8k)*, and *Suppression L31→L31 (8k)* all reach 59.5% accuracy but shorten outputs by 1000–3000 tokens on average. Steering can recover baseline-level accuracy with reduced generation length, highlighting layer-specific suppression and light amplification as especially effective strategies. "Applied layers" lists the edited layers; "Vector source" indicates the layer used to compute u; "Per-layer (own)" means each layer used its own u_ℓ.

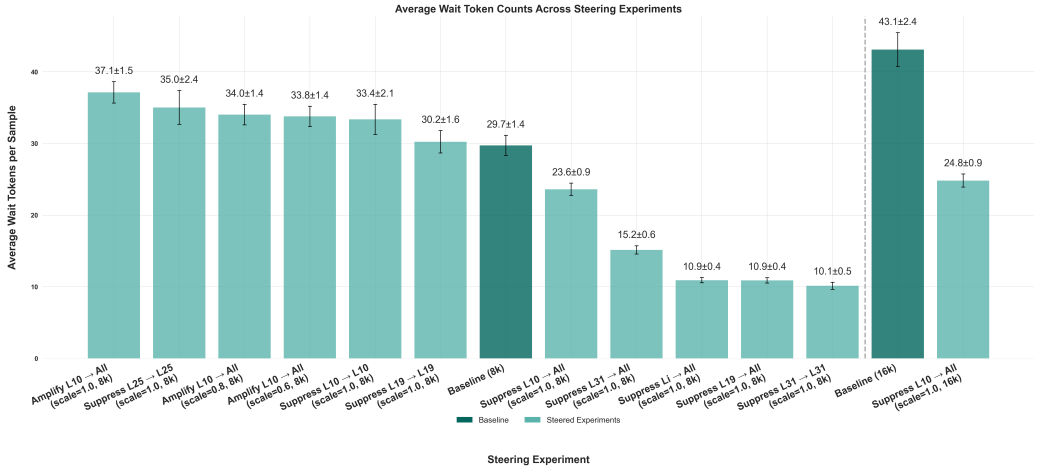

Figure 2: Suppressing backtracking directions removes unnecessary detours in reasoning traces, substantially cutting both backtracking frequency and overall output length without degrading task accuracy.

5 Future Work

Our experiments show that simple linear interventions modestly improve the accuracy–efficiency tradeoff in reasoning models, though many directions remain for future work.

First, we plan to evaluate our steering interventions out-of-domain, using new datasets. This will clarify whether the observed gains are genuine improvements in reasoning robustness or artifacts of in-domain adaptation. Second, we will test robustness under prompt perturbations.

A recurring theme in our work log is that steering vectors often encode a mixture of behaviors. For example, a backtracking direction may contain both helpful corrections and unhelpful hesitation. Rather than fully suppressing against all other discovered vectors, we aim to propose an oblique projection method that selectively subtracts away overlapping components with "undesirable" directions while preserving overlap with "helpful" directions, allowing for more fine-grained control over causal subspaces in model activations.

References

Anonymous. Thoughtology: Understanding and controlling the thoughts of reasoning llms. *arXiv preprint arXiv:2408.03314*, 2024.

Anonymous. Stop overthinking: A survey on efficient reasoning for large language models. *ResearchGate preprint*, 2025. URL https://www.researchgate.net/publication/390038709_Stop_Overthinking_A_Survey_on_Efficient_Reasoning_for_Large_Language_Models.

DeepSeek-AI. Deepseek-r1: Scaling up reasoning via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Gabriel Ilharco et al. Editing models with task vectors. *arXiv preprint arXiv:2212.04089*, 2022.

Open R1 Project. Openr1-math-220k: A large-scale dataset for mathematical reasoning. open-r1/OpenR1-Math-220k on Hugging Face, 2025. Generated by DeepSeek R1 with 2–4 verified reasoning traces per problem; described in Hugging Face dataset card and Open R1 blog update; licensed under Apache 2.0.

Sathya Rajamanoharan et al. Representation engineering: A top-down approach to steer large language models. *arXiv preprint arXiv:2501.01719*, 2025.

Thomas Venhoff et al. Steering vectors for backtracking in thinking llms. *arXiv preprint arXiv:2506.18167*, 2025. URL https://arxiv.org/abs/2506.18167.

Xuezhi Wang et al. Self-consistency improves chain of thought reasoning in language models. In *ICLR*, 2023.

Jason Wei et al. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.

A Appendix

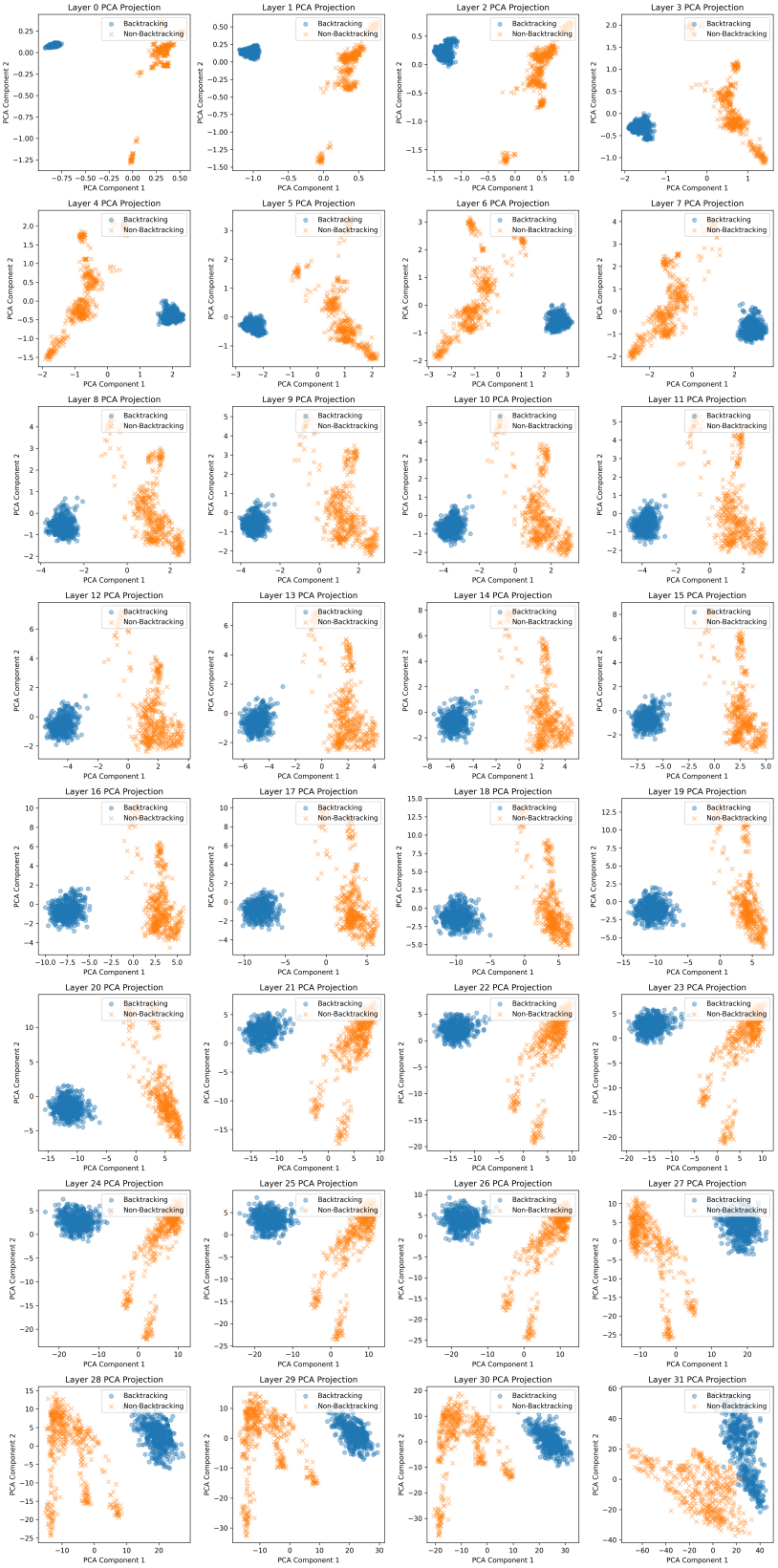

Figure 3: PCA projections of hidden activations for backtracking vs. non-backtracking traces across all layers. Each panel shows the 2D PCA projection of activations at a given layer ℓ. These visualizations illustrate the separation that underlies the steering vectors u_ℓ described in Section 3.1.

Category	Example Cues
Backtracking	"Wait", "Let's reconsider", "On second thought", "I made a mistake", "Let's try again"
Deduction	"Therefore", "This implies", "Hence", "Thus", "It follows that"
Verification	"Let me check", "verifying", "rechecking", "confirming", "This simplifies to"
Uncertainty	"I think", "maybe", "probably", "not sure", "this might be"
Step-by-step	"First", "Then", "Next", "Finally", "Let's break it down"
Heuristic shortcut	"quick trick", "shortcut", "we can estimate", "approximate"

Table 2: List of cue phrases used to identify and categorize reasoning patterns from sentence onsets.

A.1 Metrics: Additional Details

A.1.1 Backtracking Rate

Given a generated output $\hat{x}$, let $S(\hat{x})$ be the set of sentences (segmented either with spaCy rules or a simple period/newline heuristic). We compute the backtracking rate as the fraction of sentences whose first tokens match a cue in the lexicon from Table 2:

$$\mathrm{BacktrackRate} = \frac{1}{|S(\hat{x})|} \sum_{s \in S(\hat{x})} \mathbb{1}\{\mathrm{prefix}(s) \in \mathcal{C}\}.$$

A.1.2 Token–Accuracy Pareto

To capture the efficiency trade-off between correctness and verbosity, we also analyze the empirical Pareto frontier of accuracy versus output length. A method A dominates B if $\mathrm{Acc}(A) > \mathrm{Acc}(B)$ and $\mathrm{Len}(A) < \mathrm{Len}(B)$.