Triggering Chain-of-Thought via Latent Feature Interventions in Large Language Models

Zhenghao He, Guangzhi Xiong, Bohan Liu, Sanchit Sinha, Aidong Zhang
University of Virginia, Virginia, USA
{zhenghao, guangzhi, qzp4ta, sanchit, aidong}@virginia.edu

Abstract

Chain-of-Thought (CoT) prompting often improves the reasoning performance of large language models (LLMs), but the internal signal that triggers this behavior remains poorly understood. Leveraging the sparse features captured by Sparse Autoencoders (SAEs), we propose a systematic framework to analyze and intervene on the internal representations of LLMs, identifying a small set of latent features that are linked to reasoning behavior and can be causally tested through targeted intervention. Across multiple model families and reasoning benchmarks, we show that steering one or a small number of reasoning-related latent features can substantially induce reasoning behavior without explicit CoT prompting, achieving accuracy comparable to CoT. We further show that the identified features are not tied to particular wording patterns or verbosity, and confirm their causal role in reasoning through suppression experiments that impair performance even under CoT prompting. These results suggest that CoT prompting activates specific latent features to trigger reasoning, and that targeted intervention on these features offers an alternative pathway to elicit efficient reasoning behavior without explicit CoT prompting. Code is available at https://github.com/Zhenghao-He/LatentCoT.

1 Introduction

Chain-of-Thought (CoT) prompting has become a standard technique for improving reasoning in large language models, with consistent gains across arithmetic (Lewkowycz et al., 2022), symbolic, and logical reasoning tasks (Wei et al., 2022; Kojima et al., 2022; Talmor et al., 2019; Wang et al., 2022). Yet a fundamental question remains open: *is CoT prompting the unique mechanism that enables multi-step reasoning, or is it merely one of several ways to activate a latent reasoning capability?* If reasoning corresponds to an internal computational state, it should in principle be possible to activate this state through means other than explicit step-by-step prompting (Figure 1).

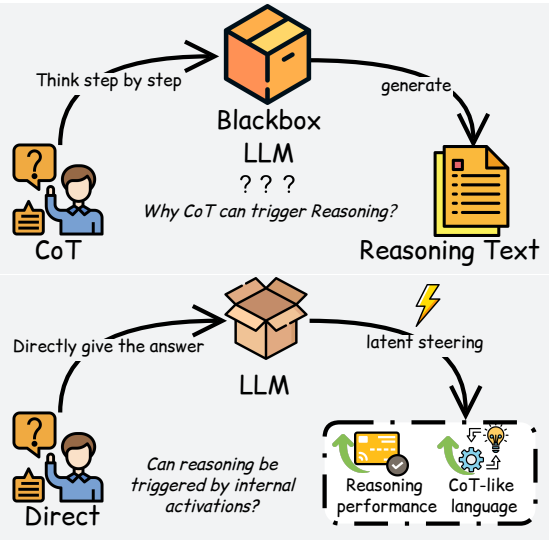


Figure 1: **Multiple triggers for latent reasoning in LLMs.** *Top:* CoT prompting produces explicit reasoning text, while the internal mechanism responsible for its effectiveness remains unclear. *Bottom:* We view reasoning as a latent internal mechanism that can be activated through different triggers, including latent steering.

Recent studies provide preliminary evidence for this view. Reasoning-relevant trajectories can be elicited by modifying the decoding process without explicit prompting (Wang and Zhou, 2024), and continuous "soft thought" representations can enhance reasoning without generating step-by-step text (Xu et al., 2025). Moreover, causal analyses suggest that CoT traces are not necessarily the mechanism that produces the final answer (Bao et al., 2024). However, these approaches either operate at the output level or lack direct causal evidence linking specific internal representations to reasoning behavior.

In this work, we analyze and intervene on the internal representations of large language models

to investigate this question. A natural starting point is to steer the model along the mean activation difference between CoT and direct prompting, following standard representation engineering approaches (Tang et al., 2025; Ma et al., 2025; Sheng et al., 2025; Lin et al., 2025; Venhoff et al., 2025). However, such dense steering directions entangle reasoning-relevant signals with formatting, verbosity, and other prompt-dependent artifacts, limiting both their effectiveness and cross-task generalization. To address this, we employ Sparse Autoencoders (SAEs) to decompose model activations into disentangled latent features, enabling targeted intervention on dimensions associated with reasoning. This design choice is empirically validated: across all models tested, SAE-based sparse feature steering substantially outperforms dense steering, and in some cases dense steering fails entirely while our method produces large accuracy gains (Section 4.3).

Using this approach across three model families and six models, we find that steering one or a small number of latent features at the first generation step can induce coherent reasoning behavior under direct prompting. We further show that the identified features are not tied to particular wording patterns or formatting (Section 4.5), and that these features are causally involved in reasoning, as suppressing them substantially impairs performance even under CoT prompting (Section 4.4). Together, these results suggest that CoT prompting activates specific latent features to trigger reasoning, and that targeted intervention on these features offers an alternative pathway to elicit reasoning without explicit CoT prompting.

Our main contributions are:

- We propose a two-stage framework using SAEs to identify latent features associated with reasoning, and show that SAE-based steering substantially outperforms dense steering, highlighting the necessity of disentangled representations.
- Across six models from three families (up to 70B) and three benchmarks, we show that steering the identified features induces reasoning behavior under direct prompting, achieving accuracy comparable to CoT while producing shorter outputs.
- Through multiple controlled experiments, we provide evidence that the identified features are causally linked to reasoning computation, generalize across tasks and prompting styles, and are not reducible to surface formatting patterns.

2 Related Work

Chain-of-Thought and Multi-step Reasoning. Chain-of-thought (CoT) prompting has been widely adopted as a practical approach for improving performance on tasks that require multi-step reasoning (Wei et al., 2022). Prior work has demonstrated that encouraging models to produce intermediate reasoning steps can lead to substantial gains across a variety of domains, including arithmetic problem solving (Lewkowycz et al., 2022), symbolic manipulation, and logical inference (Talmor et al., 2019). As a result, CoT-style prompting has become a common component in reasoning benchmarks and evaluation protocols for large language models, and has inspired numerous extensions such as self-consistency (Wang et al., 2022), structured reasoning prompts (Kojima et al., 2022), and methods that improve reasoning through attention-guided adaptation (Sinha et al., 2026).

Reasoning Beyond Explicit CoT Prompting. Beyond the standard CoT prompting paradigm, a growing line of work suggests that multi-step reasoning behavior in large language models need not be uniquely tied to explicit CoT prompts. For instance, recent studies show that CoT-style reasoning trajectories can be elicited by altering the decoding process without using explicit prompting (Wang and Zhou, 2024), and that implicit reasoning leveraging internal hidden states can support complex reasoning without generating step-by-step text (Deng et al., 2023). Moreover, methods based on continuous or latent representations, such as soft thought tokens, demonstrate enhanced reasoning capability without relying on explicit verbal reasoning steps (Xu et al., 2025). Complementary empirical analyses further indicate that the effectiveness of CoT prompting does not strictly depend on correct or valid intermediate chains, suggesting that the internal drives for reasoning extend beyond the surface verbal structure (Wang et al., 2023).

Internal Representations and Activation Steering for Reasoning. Beyond output-level analyses, prior work studies reasoning through internal representations. Probing, activation analysis, causal tracing, and concept-based representation analysis associate hidden states with meaningful behaviors and concepts (Burns et al., 2022; Meng et al., 2022; He et al., 2025). In reasoning settings, such analyses suggest that substantial computation can occur internally even without explicit verbalization.

Activation-steering methods provide a comple-

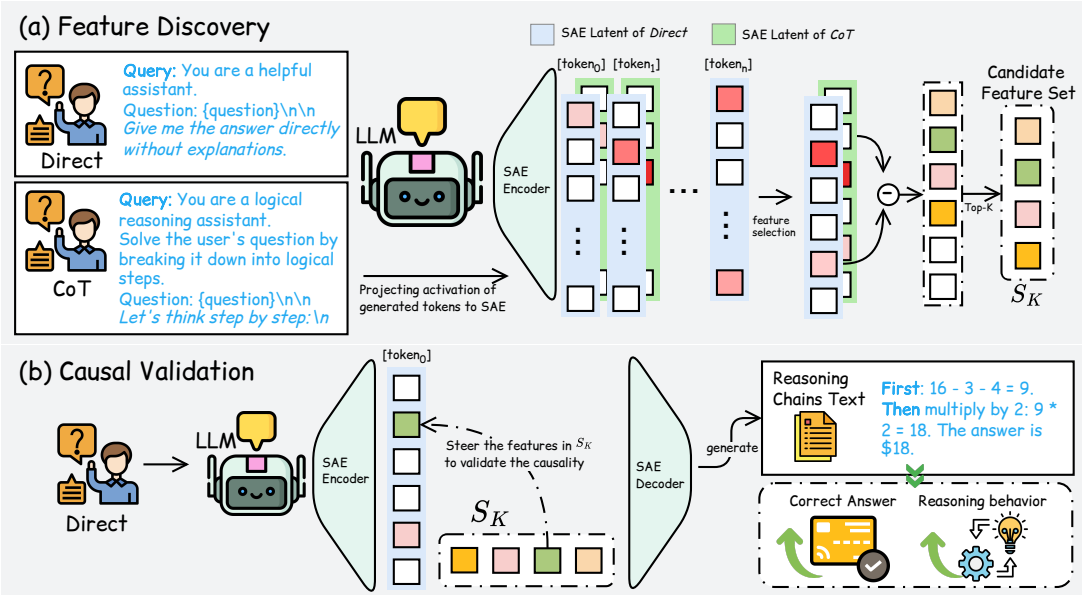


Figure 2: **Overview of the proposed two-stage pipeline.** (a) *Feature Discovery.* We contrast direct and chain-of-thought prompting to extract token-level activations, project them into sparse latent features using a pretrained sparse autoencoder (SAE), and identify prompt-sensitive candidate features via differential analysis. (b) *Causal Validation.* We apply targeted latent steering to selected features and inject the resulting residual into the model to assess their intervention sensitivity, evaluating the effect on model behavior and answer correctness.

mentary approach for controlling model behavior through internal representations. Methods such as ActAdd (Turner et al., 2024) and Contrastive Activation Addition (CAA) (Rimsky et al., 2024) construct dense steering directions from contrastive prompts or examples. While effective for behavioral control, such directions may entangle the target behavior with other activation differences induced by the contrastive conditions.

Sparse autoencoders (SAEs) instead decompose model activations into sparse latent features, enabling more localized and interpretable interventions (Huben et al., 2024). Sparse latent representations and dictionary-learning approaches have been used for concept discovery, interpretation, and intervention across language and vision models (Xiong et al., 2026; Liu et al., 2026; He et al., 2026), including in reasoning-related settings (Wang et al., 2025). Galichin et al. (2026) identify reasoning-related SAE features in reasoning models and examine their steering effects. Fang et al. (2026) use SAE features to control fine-grained reasoning strategies such as backtracking and cross-verification, while Zhang et al. (2025) discover behaviors such as reflection and backtracking from step-level reasoning activations.

These studies primarily analyze or control representations within an already-active reasoning process. In contrast, we study whether sparse latent features can causally trigger reasoning under direct prompting, before any CoT trace is generated.

3 Method

In this section, we present a two-stage pipeline for identifying and intervening on latent features associated with reasoning-related behaviors in large language models. Chain-of-Thought prompting is used as a contrasting condition to reveal prompt-dependent differences in latent activations. As illustrated in Figure 2, we first extract sparse latent features using a pretrained sparse autoencoder (SAE) and identify candidate features (Section 3.3). We then define a latent steering procedure to estimate the intervention sensitivity of individual features on training data, and evaluate the selected features on a held-out test set (Section 3.4).

3.1 Problem Setup

We consider a pretrained large language model f_θ with fixed parameters. Given an input question x and a prompting condition p, the model generates an output sequence autoregressively. We focus on two prompting regimes: direct prompting p^{dir} and chain-of-thought prompting p^{CoT}, which differ only in their instructions:

$$y \sim f_\theta(x, p), \quad p \in \{p^{\text{dir}}, p^{\text{CoT}}\}. \tag{1}$$

Let $h_{l,t}^{(p)} \in \mathbb{R}^d$ denote the hidden activation at layer l and token position t under prompting condition p, where

$$h_{l,t}^{(p)} = f_\theta^{(l)}(x, p, y_{<t}). \tag{2}$$

Our analysis focuses on latent activations aggregated from the generation process, with the specific aggregation strategy selected empirically and described in later sections.

3.2 Latent Feature Extraction with SAE

To obtain a sparse and intervention-friendly latent representation of model activations, we employ a pretrained SAE. For a given hidden activation $h_{l,t}^{(p)} \in \mathbb{R}^d$ at layer l and token position t, the SAE encoder maps it to a latent vector $z_{l,t}^{(p)} \in \mathbb{R}^m$:

$$z_{l,t}^{(p)} = \phi(W_{\text{enc}} h_{l,t}^{(p)} + b_{\text{enc}}), \tag{3}$$

where $\phi(\cdot)$ denotes an element-wise activation function, depending on the specific SAE variant. The corresponding decoder reconstructs the activation as

$$\hat{h}_{l,t}^{(p)} = W_{\text{dec}} z_{l,t}^{(p)} + b_{\text{dec}}. \tag{4}$$

The SAE is trained to minimize a reconstruction objective of the form

$$\mathcal{L}_{\text{SAE}} = \mathbb{E}\left[\left\| h_{l,t}^{(p)} - \hat{h}_{l,t}^{(p)} \right\|_2^2\right] + \lambda\, \Omega\left(z_{l,t}^{(p)}\right), \tag{5}$$

where $\Omega(\cdot)$ denotes a sparsity-inducing regularizer on the latent activations, such as an ℓ_1 penalty or a top-k masking constraint, depending on the SAE variant. In this work, we use a fixed, pretrained SAE and keep its parameters frozen during the analysis.

3.3 Differential Feature Discovery

We identify candidate latent features by contrasting their activation statistics under different prompting regimes. For a fixed input distribution over questions $x \sim \mathcal{D}$, direct prompting and chain-of-thought prompting induce two conditional distributions over latent activations.

Feature Aggregation. Given a single generated sequence under the prompting condition p, let $\{z_{l,t}^{(p)}(x)\}_{t=1}^T$ denote the SAE latent activations extracted at layer l across token positions. We first apply an aggregation function $g(\cdot)$ to obtain a fixed-dimensional representation for each example:

$$\tilde{z}^{(p)}(x) = g\left(\{z_{l,t}^{(p)}(x)\}_{t=1}^T\right), \tag{6}$$

where $g(\cdot)$ denotes a fixed aggregation over token-level activations. In our main experiments, we select the latent representation at the first generation step. The rationale for first-step intervention is analyzed in Appendix A.10. We also evaluated alternative aggregation strategies, including average/max pooling over the generated sequence, but found them to be consistently less effective.

We hypothesize that latent features associated with CoT-style behavior are activated early in generation, before the model converges toward the final answer. Aggregating over later tokens may therefore dilute these signals, consistent with prior findings that reasoning-related activations emerge early in the generation process (David, 2025).

Candidate Feature Set. We then define the conditional mean activation of latent feature k under prompting condition p as the empirical average of its aggregated activation across a dataset of inputs:

$$\mu_k(p) = \mathbb{E}_{x\sim\mathcal{D}}\left[\tilde{z}_k^{(p)}(x)\right] = \frac{1}{|\mathcal{D}|}\sum_{x\in\mathcal{D}} \tilde{z}_k^{(p)}(x). \tag{7}$$

Using these statistics, we compute a feature-wise differential score that captures prompt-induced modulation:

$$\Delta z_k = \mu_k(p^{\text{CoT}}) - \mu_k(p^{\text{dir}}). \tag{8}$$

We select latent features that are most sensitive to changes in the prompting strategy by choosing the K dimensions with the largest absolute differential scores. Formally, we define the selected feature set as

$$S_K = \arg\max_{S\subseteq\{1,\ldots,m\},\,|S|=K} \sum_{k\in S} |\Delta z_k|\,. \tag{9}$$

This procedure identifies a compact set of prompt-sensitive latent features, without introducing additional modeling assumptions or learned probes.

3.4 Causal Validation via Latent Steering

Given the candidate features S_K identified in Section 3.3, we define a latent steering procedure for a controlled intervention on model activations.

Let $a_{l,t}(x) \in \mathbb{R}^m$ denote the pre-activation latent representation produced by the SAE encoder at layer l and generation step t. The corresponding post-activation latent is given by $z_{l,t}(x) = \phi(a_{l,t}(x))$. For a chosen intervention set $S \subseteq S_K$, we define an intervention by modifying the corresponding pre-activation dimensions:

Model	Strategy	GSM8K (Cobbe et al., 2021)		GPQA (Rein et al., 2024)		BBH (Suzgun et al., 2023)	
		Acc. (↑)	#Tok (↓)	Acc. (↑)	#Tok (↓)	Acc. (↑)	#Tok (↓)
LLaMA-3.1-8B-Instruct (Grattafiori et al., 2024)	Direct	24.5	17$_{\pm 34}$	28.7	2$_{\pm 0}$	50.8	3$_{\pm 0}$
	Steered Direct	73.3	83 $_{\pm 53}$	28.2	43$_{\pm 103}$	61.6	53$_{\pm 29}$
	CoT	79.3	234$_{\pm 70}$	20.7	423$_{\pm 98}$	77.2	146$_{\pm 28}$
	Steered CoT	84.1	227$_{\pm 68}$	22.2	414$_{\pm 99}$	78.4	146$_{\pm 27}$
LLaMA-3.3-70B-Instruct (Meta, 2024)	Direct	46.7	12$_{\pm 22}$	41.9	2$_{\pm 0}$	94.0	7$_{\pm 7}$
	Steered Direct	88.8	53$_{\pm 30}$	44.9	47$_{\pm 120}$	93.6	40$_{\pm 31}$
	CoT	96.1	268$_{\pm 63}$	19.7	495$_{\pm 41}$	100.0	226$_{\pm 44}$
	Steered CoT	96.5	257$_{\pm 66}$	18.6	474$_{\pm 66}$	100.0	228$_{\pm 39}$
Qwen3-0.6B (Yang et al., 2025)	Direct	7.9	14$_{\pm 15}$	26.7	7 $_{\pm 4}$	39.6	13$_{\pm 3}$
	Steered Direct	60.6	357$_{\pm 124}$	24.7	237$_{\pm 224}$	66.4	301$_{\pm 309}$
	CoT	59.7	400$_{\pm 108}$	15.6	508$_{\pm 18}$	82.8	537$_{\pm 431}$
	Steered CoT	59.6	393$_{\pm 109}$	16.7	506$_{\pm 24}$	78.4	559$_{\pm 487}$
Qwen3-4B (Yang et al., 2025)	Direct	34.5	32$_{\pm 25}$	32.8	8$_{\pm 5}$	83.2	13$_{\pm 2}$
	Steered Direct	60.0	141$_{\pm 127}$	27.7	418$_{\pm 171}$	88.4	74$_{\pm 138}$
	CoT	61.1	421$_{\pm 101}$	18.8	510$_{\pm 12}$	98.8	458$_{\pm 328}$
	Steered CoT	62.5	422$_{\pm 101}$	19.2	511$_{\pm 10}$	97.6	528$_{\pm 348}$
Gemma-3-4B-Instruct (Team et al., 2025)	Direct	8.4	6$_{\pm 17}$	30.8	3$_{\pm 0}$	59.6	5$_{\pm 0}$
	Steered Direct	74.0	243$_{\pm 158}$	21.7	353$_{\pm 188}$	65.6	191$_{\pm 203}$
	CoT	78.1	193$_{\pm 92}$	26.2	291$_{\pm 219}$	50.0	161$_{\pm 155}$
	Steered CoT	82.8	232$_{\pm 104}$	19.2	415$_{\pm 147}$	80.8	187$_{\pm 122}$
Gemma-3-12B-Instruct (Team et al., 2025)	Direct	18.2	5$_{\pm 2}$	32.3	2$_{\pm 0}$	86.4	4$_{\pm 0}$
	Steered Direct	70.9	101$_{\pm 98}$	33.3	18$_{\pm 26}$	95.6	36$_{\pm 18}$
	CoT	92.7	181$_{\pm 75}$	24.4	404$_{\pm 133}$	95.2	136$_{\pm 70}$
	Steered CoT	92.8	168$_{\pm 71}$	22.2	400$_{\pm 137}$	96.8	120$_{\pm 58}$

Table 1: Reasoning performance across models and benchmarks under different strategies. For each model, we compare direct prompt, Chain-of-Thought (CoT) prompting, and their steered variants. Latent steering is applied only at the first generation step along an identified reasoning-related feature, with all subsequent decoding steps left unchanged. Accuracy (%) and average number of generated tokens (#Tok) are reported for each setting.

$$\tilde{a}^{\text{steer}}_{l,t,k}(x;S) = \begin{cases} a_{l,t,k}(x) + \alpha\, \mathbb{E}[|a_{l,t,k}(x)|]\,, & k \in S, \\ a_{l,t,k}(x), & k \notin S, \end{cases} \tag{10}$$

where α is a scalar steering coefficient that controls the intervention strength.

We adopt an additive intervention on pre-activation latents to externally activate features, rather than amplify their current values, allowing the intervention to act even when a feature is inactive. The perturbation is normalized by the feature's empirical activation scale, ensuring comparable intervention strength across models.

The steered latent representation is obtained as

$$\tilde{z}^{\text{steer}}_{l,t}(x) = \phi\big(\tilde{a}^{\text{steer}}_{l,t}(x;S)\big)\,. \tag{11}$$

The modified latent representation $\tilde{z}^{\text{steer}}_{l,t}(x)$ is mapped back to the activation space using the SAE decoder,

$$\hat{h}^{\text{steer}}_{l,t}(x) = W_{\text{dec}}\tilde{z}^{\text{steer}}_{l,t}(x) + b_{\text{dec}}. \tag{12}$$

Because the sparse autoencoder produces imperfect reconstruction, directly injecting the decoded steered representation may introduce reconstruction bias. Therefore, we apply a residual injection scheme. Let $\hat{h}_{l,t}(x) = \text{Dec}(z_{l,t}(x))$ denote the reconstruction of the original activation. The activation injected into the model is given by

$$\tilde{h}_{l,t}(x) = h_{l,t}(x) + \Big(\hat{h}^{\text{steer}}_{l,t}(x) - \hat{h}_{l,t}(x)\Big)\,. \tag{13}$$

This residual correction ensures a localized intervention that isolates the effect of latent steering with minimal side effects.

Rather than intervening on all candidate features simultaneously, we estimate the intervention sensitivity of individual features on a held-out training set by restricting to singleton interventions $S = \{k\}$ for each $k \in S_K$. For each candidate feature, we apply the latent steering procedure and measure the resulting change. Features with consistently positive intervention sensitivity are selected. The features are selected based on the training data and are fixed during evaluation.

4 Experiments

4.1 Implementation Details

For each model, we identify reasoning-related SAE features at a single mid-to-late transformer layer and perform steering at the same layer throughout all experiments. The steering layer, feature index, and strength α are selected on a validation split (Section 3.4), fixed per model, and used consistently across all datasets. Model-specific configurations are provided in Appendix A.6. Unless otherwise specified, accuracy is reported from a single evaluation run with fixed decoding settings, while generated-token statistics are reported as the mean and standard deviation across evaluation examples.

Datasets. We evaluate our method on three reasoning benchmarks: GSM8K (Cobbe et al., 2021), GPQA (Rein et al., 2024), and Big-Bench Hard (BBH) (Suzgun et al., 2023). For each base model, reasoning-related features are identified using 1,000 question–answer pairs sampled from the *training split of GSM8K*. All reported evaluation results are obtained by steering the identified features on the GSM8K test set, the GPQA Diamond subset, and the logical_deduction_three_objects task from Big-Bench Hard (BBH).

Models. We study six language models of varying sizes: LLaMA-3.1-8B-Instruct, LLaMA-3.3-70B-Instruct, Qwen-3-0.6B, Qwen-3-4B, Gemma-3-4B-Instruct, and Gemma-3-12B-Instruct. All models are used in inference-only mode, with weights frozen throughout all experiments.

Sparse Autoencoders. We employ pretrained SAEs released by Goodfire (Balsam et al., 2025) for LLaMA models and Gemma Scope 2 (McDougall et al., 2025) for Gemma models. For Qwen models, we train SAEs from scratch; training details are provided in Appendix A.4.

4.2 Effects of Latent Steering on Reasoning Performance

For each base model, we first identify a small set of reasoning-related latent features using the procedure described in Section 3.3 and Section 3.4. The specific feature indices selected for each model and the corresponding α are reported in Appendix A.5.

During evaluation, we apply latent steering only at the *first generation step* along the identified reasoning feature, following the step-wise intervention strategy introduced in Section 3.4. All other model components, decoding parameters, and prompts are kept identical to the unsteered baselines. The resulting performance is shown in Table 1.

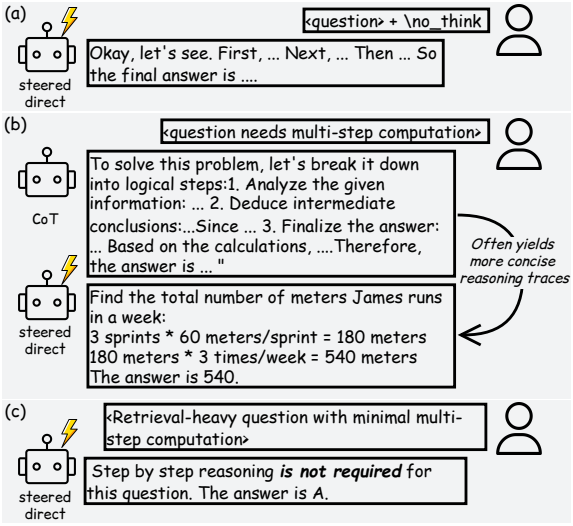


Figure 3: **Behaviors induced by steering.** (a) Steering can override prompt-level instructions that suppress reasoning (e.g., `\no_think`). (b) For questions requiring multi-step computation, steering can yield shorter explicit reasoning traces compared to standard chain-of-thought prompting. (c) On retrieval-heavy questions, latent steering leads the model to explicitly note that step-by-step reasoning is unnecessary. *These examples are illustrative and not intended to be exhaustive.*

Steering improves reasoning accuracy on reasoning-intensive tasks. Compared to direct prompting, steered direct yields large accuracy gains on reasoning-intensive benchmarks such as GSM8K and BBH. This indicates that the identified latent features are closely tied to internal reasoning processes. Notably, steering can induce reasoning behavior even when the prompt explicitly discourages intermediate reasoning. For example, on Qwen models, steered direct decoding produces coherent multi-step reasoning despite prompts containing the control token `\no_think`, which is designed to suppress the explicit generation of CoT (Figure 3a). This behavior suggests that the intervention operates at the level of internal computation rather than surface prompt compliance.

Early single-feature steering leads to more concise reasoning outputs in large models. For larger models such as LLaMA-3.3-70B, steering achieves accuracy comparable to or exceeding standard CoT prompting while substantially reducing the length of generated reasoning text (Figure 3b). Notably, this is achieved by steering a single latent feature (feature #13709) at only the first generation step.

Steering provides limited benefit once the model

is already in a reasoning mode. Steered CoT yields only marginal improvements over standard CoT. We hypothesize that when the reasoning-related feature is already strongly activated under CoT prompting, additional amplification does not further enhance reasoning performance. This observation is consistent with the view that steering primarily acts as a trigger for entering a reasoning mode, rather than refining an ongoing reasoning process, as further analyzed in Appendix A.11.

Limited gains on tasks with weak reliance on multi-step reasoning. On GPQA, where CoT prompting does not consistently outperform direct decoding, steering likewise provides little or no improvement. In some cases, models refuse to do reasoning and directly produce answers when we strongly activate those reasoning features (Figure 3c). This suggests that when a task does not strongly rely on multi-step reasoning, the identified features are less engaged and steering has limited effect. This is supported by Section 4.7, where features identified from GPQA fail to improve GPQA itself but improve the other two benchmarks.

4.3 Comparison with Dense Steering Baseline

To compare our method with standard representation engineering approaches, we construct a dense CoT steering direction using the mean activation difference between CoT and direct prompting. Specifically, we extract the hidden state of the last prompt token at the same layer used for SAE feature discovery, and define the dense direction as the average difference between CoT and direct representations on the GSM8K training set. During generation, the vector is added to the hidden state with a fixed steering strength α, tuned on a validation split. Detailed implementation details are provided in Appendix A.7. The resulting direction is then evaluated on GSM8K test and BBH.

As shown in Table 2, dense steering yields smaller improvements than SAE feature steering across all models. Moreover, the dense direction transfers poorly to BBH, whereas SAE steering consistently improves reasoning performance across both datasets. We also observe that dense steering is ineffective on Qwen3, where prompts include explicit <no_think> tags. One possible explanation is that the dense direction mainly captures dataset-specific activation differences between CoT and direct prompting. As a result, it may act as a task-dependent optimization direction and generalize poorly across tasks or prompting formats.

Model	Strategy	GSM8K (Cobbe et al., 2021)		BBH (Suzgun et al., 2023)	
		Acc. (↑)	#Tok	Acc. (↑)	#Tok
LLaMA-3.1 -8B-Instruct (Grattafiori et al., 2024)	Direct	24.5	17 $\pm$34	50.8	3 $\pm$0
	Dense	<u>68.4</u>	88 $\pm$65	<u>56</u>	3 $\pm$0
	SAE	**73.3**	83 $\pm$53	**61.6**	53 $\pm$29
Qwen3-4B (Yang et al., 2025)	Direct	34.5	32 $\pm$25	83.2	13 $\pm$2
	Dense	<u>34.6</u>	32 $\pm$29	<u>83.6</u>	13 $\pm$2
	SAE	**60.0**	141 $\pm$127	**88.4**	74 $\pm$138
Gemma-3 -4B-Instruct (Team et al., 2025)	Direct	8.4	6 $\pm$17	59.6	5 $\pm$0
	Dense	<u>16.2</u>	46 $\pm$118	<u>59.6</u>	5 $\pm$0
	SAE	**74.0**	243 $\pm$158	**65.6**	191 $\pm$203

Table 2: Comparison between dense representation steering and sparse SAE feature steering under direct prompting. **Bold** indicates the best result for each model, while <u>underlined</u> values denote the second-best results.

In contrast, SAE isolates a sparse latent feature that more cleanly captures the reasoning signal, enabling more reliable intervention.

4.4 Suppression under CoT Prompting

We further examine the causal role of the identified features by suppressing their contributions during CoT prompting. Using Qwen3-4B, we intervene on the top-25 identified reasoning-related features at layer 29 throughout both prompt processing and generation. At each forward pass, we project the positive activations of the target SAE features back into the model hidden space using their corresponding decoder directions, and subtract a scaled version of this contribution from the residual stream. The intervention magnitude is calibrated on held-out validation data for each dataset, while all other generation settings are kept unchanged.

As shown in Table 3, suppression substantially reduces CoT accuracy on both GSM8K and BBH. This cross-dataset degradation provides additional causal evidence that the identified features contribute to reliable reasoning under CoT prompting.

Dataset	Strategy	Acc. (%)	Avg. Tokens
GSM8K	CoT	61.1	421
	CoT + Supp.	12.1	479
BBH	CoT	98.8	458
	CoT + Supp.	62.8	401
GPQA	CoT	18.8	510
	CoT + Supp.	21.7	476

Table 3: Effect of suppressing the hidden-state contributions of the top-25 reasoning-related SAE features under CoT prompting using Qwen3-4B.

Importantly, suppression does not cause generation length to collapse. The suppressed outputs remain comparable in length to standard CoT re-

sponses rather than reverting to short direct-answer generations. Thus, the model can continue to produce extended responses while its answer accuracy substantially deteriorates, suggesting that long-form generation alone is not sufficient for reliable reasoning.

GPQA exhibits a different pattern: suppression slightly improves over the CoT baseline. This is consistent with our earlier observation that CoT prompting itself provides limited benefit on GPQA.

4.5 Beyond Prompt Wording and Output Verbosity

A natural concern is whether the identified features simply track specific prompt wording or output verbosity rather than reasoning itself. We address this from two complementary directions.

Cross-prompt generalization. Figure 4 shows the activation of the identified reasoning feature under different prompting conditions on GSM8K. Several alternative prompts that encourage reasoning (e.g., "Explain how", "Solve carefully", "Think") consistently activate the same feature, in some cases more strongly than explicit step-by-step prompting (detailed prompts can be found in Appendix A.3). This indicates that the feature is not tied to a specific prompt template but responds to diverse linguistic cues that encourage reasoning. Notably, stronger activation does not necessarily translate into higher accuracy: despite exhibiting comparable or even stronger activation than step-by-step prompting, these alternative prompts achieve similar accuracy. This observation is consistent with our analysis in Appendix A.11, which suggests that activation of this feature is better understood as a reasoning-mode signal than as a direct predictor of answer correctness.

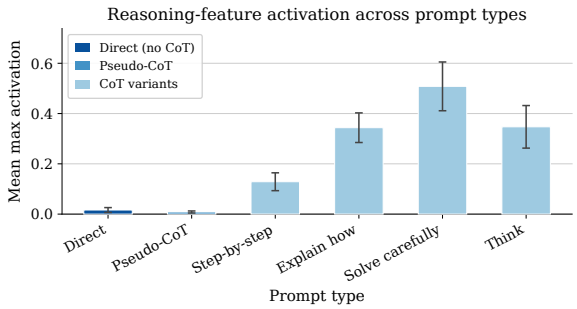


Figure 4: Activation of the reasoning-related SAE feature under different prompting conditions on GSM8K using LLaMA-3.1-8B-Instruct. Pseudo-CoT mimics step-by-step formatting without performing reasoning.

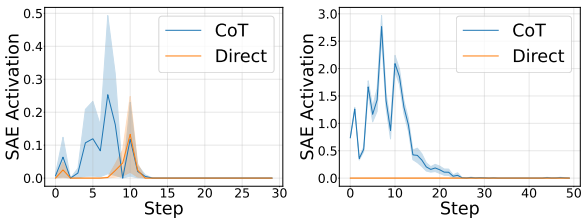
Pseudo-CoT. Different reasoning prompts also tend to produce similar discourse markers (e.g., "First", "Next"). To rule out that the feature merely tracks such formatting patterns, we construct a *pseudo-CoT* condition using the following prompt:

> "Provide a detailed restatement of the problem in a step-by-step format. Use expressions such as 'First,' 'Next,' and 'Therefore,' but do not perform any calculations or intermediate reasoning. After the restatement, give the final answer directly."

As shown in Figure 4, pseudo-CoT activates the feature even less than direct prompting, while CoT variants activate it substantially more strongly. Together, these results confirm that the feature is linked to reasoning computation, not surface formatting or verbosity.

4.6 Feature Dynamics During Generation

Figure 5 visualizes the activation dynamics of the reasoning-related SAE features used for steering during answer generation on GSM8K; these same single latent dimensions are intervened on to obtain all steered results reported in Table 1. Across both models, the identified features exhibit a highly non-uniform temporal profile, with activations peaking early in decoding and rapidly decaying thereafter. This transient pattern suggests that the feature is primarily engaged during the initial phase of reasoning rather than throughout generation.



(a) Reasoning feature #8629 in LLaMA-3.1-8B-Instruct (b) Reasoning feature #13709 in LLaMA-3.3-70B-Instruct

Figure 5: Activation dynamics of a reasoning-related SAE feature during generation on GSM8K.

Moreover, CoT prompting consistently induces stronger activation of the reasoning feature compared to direct decoding, indicating that the feature is associated with reasoning-oriented generation modes. Despite differences in scale, this temporal structure is qualitatively consistent across model sizes, with larger models exhibiting more concentrated and higher-magnitude activation. These observations support our focus on early interventions when steering reasoning-related latent features.

4.7 Cross-Dataset Feature Identification

To examine whether the identified feature is tied to a specific dataset or task, we perform cross-dataset feature identification and evaluation. Specifically, we independently extract reasoning-related features from GSM8K, GPQA, and BBH, and evaluate their steering effects across all three benchmarks.

The results are shown in Figure 6. Features identified from GSM8K and GPQA both improve performance on GSM8K and BBH. Interestingly, the feature identified from GPQA does not improve GPQA itself, but still improves performance on GSM8K and BBH. This suggests that the identified feature is not limited to arithmetic reasoning. Instead, it indicates that activating CoT-style reasoning does not substantially benefit GPQA, while the reasoning feature remains useful on tasks where step-by-step reasoning improves performance.

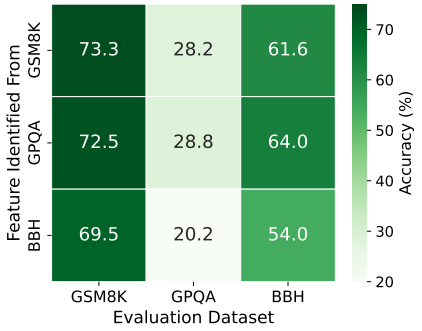


Figure 6: Cross-dataset evaluation of reasoning features identified from different datasets.

To further characterize the extent to which feature discovery is shared across datasets, we measure the overlap between the top-k candidate feature sets independently identified from each dataset. For two datasets with top-k feature sets A_k and B_k, we compute

$$\text{Jaccard@}k(A, B) = \frac{|A_k \cap B_k|}{|A_k \cup B_k|}. \qquad (14)$$

We report the results for $k = 25$ in Table 4. All dataset pairs exhibit non-zero overlap, indicating that feature discovery recovers partially shared reasoning-related SAE features rather than entirely dataset-specific sets. The overlap is highest between GPQA and BBH (0.28), consistent with the strong GPQA-to-BBH transfer observed in Figure 6. At the same time, the relatively limited overlap across datasets indicates that feature identification remains substantially dataset-dependent.

	GSM8K	GPQA	BBH
GSM8K	1.00	0.11	0.09
GPQA	–	1.00	**0.28**
BBH	–	–	1.00

Table 4: Cross-dataset overlap of the top-25 candidate reasoning-related SAE features, measured by Jaccard similarity.

5 Conclusion

In this work, we show that multi-step reasoning in large language models can be triggered by intervening on latent features identified through sparse autoencoders. Across six models from three families, steering one or a small number of features induces reasoning behavior under direct prompting, approaching CoT accuracy with shorter outputs. We further show that these features are specific to reasoning computation rather than surface formatting, and suppressing them impairs performance under CoT prompting. These findings suggest that CoT prompting activates a latent reasoning mechanism that can also be triggered through targeted internal intervention, opening a pathway toward more efficient and controllable reasoning in language models.

Limitations

Although our results are consistent with the interpretation that SAE representations partially disentangle latent features related to internal reasoning computation from those associated with verbose verbalization, we do not claim that the identified features constitute fully disentangled or isolated reasoning mechanisms. Our conclusions are based on behavioral and intervention-level evidence rather than complete mechanistic characterization. Our evaluation is limited to six models and three benchmarks, and broader validation across models and tasks remains for future work.

Acknowledgments

This work is supported in part by the US National Science Foundation (NSF) under grants IIS-2106913, IIS-2538206, IIS-2529378, CCF-2217071, and CNS-2213700. Any recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF. We also acknowledge the use of ChatGPT solely for assistance with grammar, punctuation, vocabulary, and language polishing.

References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, and 3 others. 2025. Smollm2: When smol goes big – data-centric training of a small language model.

Daniel Balsam, Thomas McGrath, Liv Gorton, Nam Nguyen, Myra Deng, and Eric Ho. 2025. Announcing open-source saes for llama 3.3 70b and llama 3.1 8b. Goodfire Research.

Guangsheng Bao, Hongbo Zhang, Linyi Yang, Cunxiang Wang, and Yue Zhang. 2024. Llms with chain-of-thought are non-causal reasoners. *CoRR*.

Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. 2022. Discovering latent knowledge in language models without supervision. *arXiv preprint arXiv:2212.03827*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Joey David. 2025. Temporal predictors of outcome in reasoning language models. *arXiv preprint arXiv:2511.14773*.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. 2023. Implicit chain of thought reasoning via knowledge distillation. *arXiv preprint arXiv:2311.01460*.

Yi Fang, Wenjie Wang, Mingfeng Xue, Boyi Deng, Fengli Xu, Dayiheng Liu, and Fuli Feng. 2026. Controllable llm reasoning via sparse autoencoder-based steering. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 21290–21305.

Andrey V Galichin, Alexey Dontsov, Polina Druzhinina, Anton Razzhigaev, Oleg Rogov, Elena Tutubalina, and Ivan Oseledets. 2026. I have covered all the bases here: Interpreting reasoning features in large language models via sparse autoencoders. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 30771–30779.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Zhenghao He, Sanchit Sinha, Guangzhi Xiong, and Aidong Zhang. 2025. Gcav: A global concept activation vector framework for cross-layer consistency in interpretability. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 614–623. IEEE.

Zhenghao He, Guangzhi Xiong, Boyang Wang, Sanchit Sinha, and Aidong Zhang. 2026. Casl: Concept-aligned sparse latents for interpreting diffusion models. *arXiv preprint arXiv:2601.15441*.

Robert Huben, Hoagy Cunningham, Logan Smith, Aidan Ewart, and Lee Sharkey. 2024. Sparse autoencoders find highly interpretable features in language models. In *International Conference on Learning Representations*, volume 2024, pages 7827–7845.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857.

Zhengkai Lin, Zhihang Fu, Ze Chen, Chao Chen, Liang Xie, Wenxiao Wang, Deng Cai, Zheng Wang, and Jieping Ye. 2025. Controlling thinking speed in reasoning models. *Advances in Neural Information Processing Systems*, 38:78300–78347.

Bohan Liu, Wenqian Ye, Guangzhi Xiong, Zhenghao He, Sanchit Sinha, and Aidong Zhang. 2026. Towards robustness against typographic attack with training-free concept localization. *Preprint*, arXiv:2607.02494.

Xinyin Ma, Guangnian Wan, Runpeng Yu, Gongfan Fang, and Xinchao Wang. 2025. Cot-valve: Length-compressible chain-of-thought tuning. *Preprint*, arXiv:2502.09601.

Callum Stuart McDougall, Arthur Conmy, János Kramár, Tom Lieberum, Senthooran Rajamanoharan, and Neel Nanda. 2025. Gemma scope 2 - technical paper.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*.

Meta. 2024. meta-llama/llama-3.3-70b-instruct. Llama-3.3-70B-Instruct model.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. 2024. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15504–15522.

Leheng Sheng, An Zhang, Zijian Wu, Weixiang Zhao, Changshuo Shen, Yi Zhang, Xiang Wang, and Tat-Seng Chua. 2025. On reasoning strength planning in large reasoning models. *Advances in Neural Information Processing Systems*, 38:62978–63012.

Sanchit Sinha, Guangzhi Xiong, Bohan Liu, Zhenghao He, and Aidong Zhang. 2026. Attention-guided fine-tuning of multimodal large language models improves chain-of-thought reasoning. *Preprint*, arXiv:2606.01558.

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. 2023. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158.

Xinyu Tang, Xiaolei Wang, Zhihao Lv, Yingqian Min, Wayne Xin Zhao, Binbin Hu, Ziqi Liu, and Zhiqiang Zhang. 2025. Unlocking general long chain-of-thought reasoning capabilities of large language models via representation engineering. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6832–6849.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. 2024. Steering language models with activation engineering. *Preprint*, arXiv:2308.10248.

Constantin Venhoff, Iván Arcuschin, Philip Torr, Arthur Conmy, and Neel Nanda. 2025. Base models know how to reason, thinking models learn when. *Preprint*, arXiv:2510.07364.

Anyi Wang, Dong Shu, Yifan Wang, Yunpu Ma, and Mengnan Du. 2025. Improving llm reasoning through interpretable role-playing steering. In *EMNLP (Findings)*, pages 731–751.

Boshi Wang, Sewon Min, Xiang Deng, Jiaming Shen, You Wu, Luke Zettlemoyer, and Huan Sun. 2023. Towards understanding chain-of-thought prompting: An empirical study of what matters. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 2717–2739.

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

Xuezhi Wang and Denny Zhou. 2024. Chain-of-thought reasoning without prompting. *Advances in Neural Information Processing Systems*, 37:66383–66409.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Guangzhi Xiong, Zhenghao He, Bohan Liu, Sanchit Sinha, and Aidong Zhang. 2026. Toward faithful retrieval-augmented generation with sparse autoencoders. In *International Conference on Learning Representations*, volume 2026, pages 112549–112577.

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

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Zhenyu Zhang, Shujian Zhang, John Lambert, Wenxuan Zhou, Zhangyang Wang, Mingqing Chen, Andrew Hard, Rajiv Mathews, and Lun Wang. 2025. Fantastic reasoning behaviors and where to find them: Unsupervised discovery of the reasoning process. *arXiv preprint arXiv:2512.23988*.

A Appendix

A.1 Dataset Details

A.1.1 GSM8K

GSM8K is a benchmark of grade-school level mathematical word problems that require multi-step numerical reasoning (Cobbe et al., 2021). Each example consists of a natural language question and a short numerical answer. Solving these problems typically involves parsing the problem description, performing a sequence of arithmetic operations, and producing a final numeric result.

In our experiments, GSM8K is used both for identifying reasoning-related latent features and for evaluating reasoning performance. Specifically, we randomly sample 1,000 examples from the GSM8K training split to identify candidate reasoning-related features using the procedure described in Section 3.3. All reported evaluation results are obtained on the GSM8K test split. Accuracy is measured as exact match between the model output and the ground-truth answer, following standard evaluation protocols.

A.1.2 GPQA

GPQA is a challenging question answering benchmark designed to assess scientific reasoning at the graduate level (Rein et al., 2024). The dataset covers multiple domains, including physics, chemistry, and biology, and is constructed to be resistant to surface-level retrieval by large language models. Questions often require combining domain knowledge with logical reasoning rather than simple factual recall.

We evaluate our method on the GPQA Diamond subset, which contains questions with verified difficulty and high annotation quality. Model performance is measured by exact match accuracy over multiple-choice answers. We note that prior work has shown that chain-of-thought prompting does not consistently improve performance on GPQA, making it a useful testbed for examining whether latent steering selectively benefits tasks that rely on multi-step reasoning.

A.1.3 Big-Bench Hard

Big-Bench Hard (BBH) is a curated subset of the BIG-Bench benchmark that focuses on tasks known to be challenging for language models (Suzgun et al., 2023). The tasks in BBH are selected based on their low performance under standard prompting and often require logical deduction, symbolic manipulation, or multi-step reasoning.

In this work, we evaluate on the `logical_deduction_three_objects` task from BBH, which requires reasoning about relational constraints among multiple entities. This task emphasizes structured logical reasoning rather than numerical computation. Model performance is evaluated using exact match accuracy on the final answer. BBH serves as a complementary benchmark to GSM8K, allowing us to test whether the identified reasoning-related features generalize beyond arithmetic reasoning to logical deduction tasks.

A.2 Model Details

We evaluate our method on six pretrained, instruction-tuned large language models drawn from three widely used model families: LLaMA-3, Qwen-3, and Gemma-3. These families differ in architecture, training data, and scale, allowing us to assess the generality of latent steering across diverse model designs. All models are used in inference-only mode, with parameters frozen throughout all experiments.

A.2.1 LLaMA-3 Family

We consider two instruction-tuned models from the LLaMA-3 family: LLaMA-3.1-8B-Instruct and LLaMA-3.3-70B-Instruct. These models represent small- and large-scale regimes within the same architectural family, enabling controlled comparisons across model size. LLaMA-3 models are widely adopted as strong open-weight baselines for reasoning tasks, and prior work has shown that chain-of-thought prompting is particularly effective for larger LLaMA models. We use pretrained sparse autoencoders released by Goodfire for both LLaMA-3 models.

A.2.2 Qwen-3 Family

We evaluate two models from the Qwen-3 family: Qwen-3-0.6B and Qwen-3-4B. These models provide additional coverage of smaller-scale language models and differ substantially from LLaMA-3 in training data composition and architectural choices. For Qwen-3 models, pretrained sparse autoencoders are not publicly available, so we train sparse autoencoders from scratch following the procedure described in Appendix A.4. Including Qwen-3 allows us to examine whether reasoning-related latent features and steering effects persist in models with more limited capacity.

A.2.3 Gemma-3 Family

We use two instruction-tuned models from the Gemma-3 family: Gemma-3-4B-Instruct and Gemma-3-12B-Instruct. Gemma-3 models are trained with different data and optimization strategies compared to LLaMA-3 and Qwen-3, offering an additional axis of architectural and training diversity. We employ pretrained sparse autoencoders from GemmaScope for both Gemma-3 models. Results on Gemma-3 help assess the robustness of latent steering across independently developed model families.

A.3 Input Prompts

We use fixed prompt templates for all experiments to ensure comparability across models and conditions. For each model family, we define a *direct* prompt and a *chain-of-thought (CoT)* prompt that differ only in whether explicit step-by-step reasoning is encouraged. Unless otherwise specified, all prompts are applied consistently across datasets, with decoding parameters held fixed. Due to the length of the prompt templates and the presence of model-specific special tokens, we present the full prompts as figures for readability.

A.3.1 LLaMA-3 Family

For models in the LLaMA-3 family, we use fixed direct and chain-of-thought prompt templates. The full prompt templates are shown in Figures 7 and 8.

Direct Prompt for LLaMA Family

```
<|begin_of_text|><|start_header_id|>system<|end_header_id|>\n\n
You are a helpful assistant.<|eot_id|>
<|start_header_id|>user<|end_header_id|>\n\n
Question: {question}\n\n
Give me the answer directly without explanations.<|eot_id|>
<|start_header_id|>assistant<|end_header_id|>\n\n
```

Figure 7: Direct prompt template for the LLaMA-3 family.

CoT Prompt for LLaMA Family

```
<|begin_of_text|><|start_header_id|>system<|end_header_id|>\n\n
You are a logical reasoning assistant. Solve the user's question by
breaking it down into logical steps.
Finally, provide the answer in the specified format.<|eot_id|>
<|start_header_id|>user<|end_header_id|>\n\n
Question: {question}\n\n
Let's think step by step:\n
    1) Analyze the given information.\n
    2) Deduce intermediate conclusions.\n
    3) Finalize the answer.\n\n
Format your final response as: 'Therefore, the answer is [your
answer].'<|eot_id|>
<|start_header_id|>assistant<|end_header_id|>\n\n"
```

Figure 8: Chain-of-thought prompt template for the LLaMA-3 family.

In addition to the direct and CoT prompts, we also consider several alternative prompts for the cross-prompt analysis in Section 4.5. These prompts implicitly encourage careful reasoning without explicitly requesting step-by-step explanations. The full templates for these prompts are shown in Figures 9–11.

Think Prompt. See Figure 9 for the full template.

Think Prompt for LLaMA Family

```
<|begin_of_text|><|start_header_id|>system<|end_header_id|>\n\n
You are a helpful assistant. Think carefully before answering.
Finally, provide the answer in the specified format.<|eot_id|>
<|start_header_id|>user<|end_header_id|>\n\n
Question: {question}\n\n
Think carefully before answering.\n\n
Format your final response as: 'Therefore, the answer is [your
answer].'<|eot_id|>
<|start_header_id|>assistant<|end_header_id|>\n\n"
```

Figure 9: Think prompt template for the LLaMA-3 family.

Explain Prompt. See Figure 10 for the full template.

Explain Prompt for LLaMA Family

```
<|begin_of_text|><|start_header_id|>system<|end_header_id|>\n\n
You are a helpful assistant. Explain how to solve the user's question.
Finally, provide the answer in the specified format.<|eot_id|>
<|start_header_id|>user<|end_header_id|>\n\n
Question: {question}\n\n
Explain how you get the answer.\n\n:
Format your final response as: 'Therefore, the answer is [your
answer].'<|eot_id|>
<|start_header_id|>assistant<|end_header_id|>\n\n"
```

Figure 10: Explain prompt template for the LLaMA-3 family.

Solve Carefully Prompt. See Figure 11 for the full template.

Solve Carefully Prompt for LLaMA Family

```
<|begin_of_text|><|start_header_id|>system<|end_header_id|>\n\n
You are a helpful assistant. Solve the user's question carefully.
Finally, provide the answer in the specified format.<|eot_id|>
<|start_header_id|>user<|end_header_id|>\n\n
Question: {question}\n\n
Solve the problem carefully.\n\n
Format your final response as: 'Therefore, the answer is [your
answer].'<|eot_id|>
<|start_header_id|>assistant<|end_header_id|>\n\n"
```

Figure 11: Solve carefully prompt template for the LLaMA-3 family.

A.3.2 Qwen-3 Family

For the Qwen-3 family, we define direct and chain-of-thought prompts analogous to those used for LLaMA-3, adapted to the instruction format expected by Qwen models. The full prompt templates are shown in Figures 12 and 13.

Direct Prompt. See Figure 12 for the full template.

Direct Prompt for Qwen Family

```
<|im_start|>system
You are a helpful assistant.<|im_end|>
<|im_start|>user
{question}
Give me the answer directly without explanations.
Give only the final answer.
Do not include reasoning, analysis, or intermediate steps.
Output a single sentence only./no_think
<|im_end|>
<|im_start|>assistant\n
```

Figure 12: Direct prompt template for the Qwen-3 family.

Chain-of-Thought Prompt. See Figure 13 for the full template.

CoT Prompt for Qwen Family

```
<|im_start|>system
You are a logical reasoning assistant. Solve the user's question by
breaking it down into logical steps. Finally, provide the answer in the
specified format.<|im_end|>
<|im_start|>user
{question}
Let's think step by step:
1) Analyze the given information.
2) Deduce intermediate conclusions
3) Finalize the answer.
Format your final response as: 'Therefore, the answer is [your
answer].'/think<|im_end|>
<|im_start|>assistant
```

Figure 13: Chain-of-thought prompt template for the Qwen-3 family.

A.3.3 Gemma-3 Family

For the Gemma-3 family, we likewise use a direct prompt and a chain-of-thought prompt that follow the instruction conventions of Gemma models. The full templates are shown in Figures 14 and 15.

Direct Prompt. See Figure 14 for the full template.

Direct Prompt for Gemma Family

```
{question}\n
Give me the answer directly without explanations.<eos>
```

Figure 14: Direct prompt template for the Gemma-3 family.

Chain-of-Thought Prompt. See Figure 15 for the full template.

CoT Prompt for Gemma Family

```
You are a logical reasoning assistant. Solve the user's question by
breaking it down into logical steps.
Finally, provide the answer in the specified format.
Question: {question}\n
Let's think step by step:
1) Analyze the given information.
2) Deduce intermediate conclusions.
3) Finalize the answer.\n
Format your final response as: 'Therefore, the answer is [your answer].
<eos>
```

Figure 15: Chain-of-thought prompt template for the Gemma-3 family.

Prompt Consistency. Across all model families, the direct and CoT prompts differ only in their instructions regarding explicit reasoning. No task-specific information or answer hints are introduced

Model	Steered Feature Index	Steering Step	Steering Strength α	Steering Layer
LLaMA-3.1-8B-Instruct (Grattafiori et al., 2024)	#8629	1	15	19
LLaMA-3.3-70B-Instruct (Meta, 2024)	#13709	1	20	50
Qwen3-0.6B (Yang et al., 2025)	#15352, #14424, #27058, #22718, #12509	1	25	19
Qwen3-4B (Yang et al., 2025)	#12714, #66391, #58281, #60128, #25787	1	70	29
Gemma-3-4B-Instruct (Team et al., 2025)	#113617, #80327, #165330, #52856, #923, #51564, #8253, #6879, #1257, #5239	1	15	29
Gemma-3-12B-Instruct (Team et al., 2025)	#6662, #7847, #97, #15521, #5100, #32828, #2616, #3397, #12059, #130	1	25	31

Table 5: Steering configurations for different models. For each model, we report the index of the steered reasoning-related feature, the decoding step at which steering is applied, the steering strength α, and the steering layer.

through prompt wording. This design allows us to attribute observed differences in behavior to latent steering and prompting conditions rather than to prompt-specific artifacts.

A.4 SAE Training Information

For the Qwen3 models, we train sparse autoencoders (SAEs) from scratch using the `sparsify` library with its default configuration. All training hyperparameters follow the library defaults unless otherwise specified.

We use the `EleutherAI/SmolLM2-135M-10B` dataset as training data for the SAEs. This dataset is a sampled subset of the SmolLM2-135M pretraining corpus (derived from the data used to train SmolLM2 as described in (Allal et al., 2025)), intended for efficient training of auxiliary components. The dataset is tokenized using the corresponding model tokenizer and processed into fixed-length chunks following the standard `chunk_and_tokenize` procedure provided by the library.

SAEs are trained on the frozen base language model in an inference-only setting. We use a batch size of 16 and train the autoencoders using the default optimization and sparsity settings. No model-specific tuning is performed beyond selecting the target layers corresponding to the Qwen3 architectures. For Qwen3-4B and Qwen3-0.6B, we train sparse autoencoders with latent dimensions of 26,624 and 32$\times$ the model hidden size, respectively. Both models use Top-K sparsity with $K = 192$.

A.5 Steering Settings

Table 5 summarizes the steering configurations used across different models. For each model, we report the index of the reasoning-related feature selected for steering, the decoding step at which the intervention is applied, and the steering strength α. These settings are held fixed across tasks and datasets for a given model, and are used consistently in all steering experiments reported in the paper.

A.6 Model-specific Configurations

All experiments use pretrained causal language models loaded from their official HuggingFace checkpoints, without any additional finetuning. Models are evaluated in inference mode with parameters frozen. We enable the output of hidden states (`output_hidden_states=True`) for all models to support activation-level analysis. Models are loaded using standard half-precision (FP16), except for Gemma models which are evaluated in full precision (FP32). Tokenizers use left padding, with the end-of-sequence token assigned as the padding token when necessary.

A.7 Dense Steering Baseline

To compare SAE feature steering with a standard representation engineering baseline, we construct a dense steering direction based on the activation difference between CoT and direct prompting.

Dense Direction Construction. Given a set of problems from the GSM8K training set, we construct two prompts for each problem: a direct prompt and a CoT prompt (e.g., "Let's think step by step"). We extract the hidden state of the last

prompt token at the same transformer layer used for SAE feature discovery.

Let $h^{(i)}_{\text{cot}}$ and $h^{(i)}_{\text{direct}}$ denote the hidden states for the i-th example under CoT and direct prompting. The dense steering direction is computed as the mean activation difference:

$$v_{\text{cot}} = \mathbb{E}_i[h^{(i)}_{\text{cot}}] - \mathbb{E}_i[h^{(i)}_{\text{direct}}].$$

Steering Procedure. During generation under direct prompting, the dense direction is added to the hidden state of the same layer:

$$h' = h + \alpha v_{\text{cot}},$$

where α controls the steering strength. The intervention is applied at the first decoding step, consistent with the SAE steering experiments.

Evaluation Setup. The dense direction is constructed using the GSM8K training set only. The resulting direction is then applied to GSM8K test and BBH without modification. For each model, the steering strength α is tuned on a validation split to achieve the best reasoning performance.

Model	Steering Strength α
LLaMA-3.1-8B-Instruct	10
Qwen3-4B	10
Gemma-3-4B-Instruct	10

Table 6: Steering strength used for dense representation steering.

A.8 Evaluation on a Large Reasoning Model

We additionally evaluate our method on a large reasoning model, DeepSeek-R1-Distill-Llama-8B (DeepSeek-AI, 2025), to examine whether the identified reasoning-related features remain effective in a model explicitly trained for long-form reasoning. We use the publicly available SAE checkpoint `andreuka18/sae-deepseek-r1-llama-8b`, using the SAE at layer 19. To construct the direct-answer setting, we prepend an empty thinking block, `<think>\n\n</think>`, before generation, which suppresses explicit reasoning and encourages the model to produce a direct answer.

Table 7 reports results on GSM8K, GPQA, and BBH. Across all three datasets, latent steering substantially improves the direct-answer baseline. On GSM8K, accuracy increases from 36.8% to 74.8%, approaching the 77.4% accuracy obtained with standard CoT generation. Similarly, on BBH, steering improves accuracy from 41.6% to 96.4%, compared with 98.4% under CoT. These results suggest that the steering effect extends to an R1-distilled reasoning model.

GPQA exhibits a somewhat different behavior. Steering improves the direct-answer accuracy from 34.3% to 41.4%, approaching the 42.9% accuracy of CoT generation. However, both steered direct and CoT generation require substantially longer outputs on GPQA, exceeding 3,500 generated tokens on average. This indicates that, for the R1-distilled model, improved GPQA performance is accompanied by substantially increased generation cost.

Dataset	Strategy	Acc. (%)	Avg. Tokens
GSM8K	Direct	36.8	66
	Steered Direct	74.8	651
	CoT	77.4	694
GPQA	Direct	34.3	105
	Steered Direct	41.4	3742
	CoT	42.9	3573
BBH	Direct	41.6	26
	Steered Direct	96.4	294
	CoT	98.4	495

Table 7: Evaluation of latent steering on DeepSeek-R1-Distill-Llama-8B. Steering consistently improves the direct-answer baseline across GSM8K, GPQA, and BBH, approaching standard CoT accuracy on all three benchmarks.

A.9 Case Study: First-Token Distribution Shift

To illustrate how early steering alters the decoding trajectory, we examine the first-token probability distribution with and without steering on GSM8K using LLaMA-3.1-8B-Instruct. Figure 16 shows the top predicted tokens under both conditions. Without steering, the model's highest-probability tokens are dominated by answer-like tokens such as numbers or symbols (e.g., "$", "16", "12"), indicating a direct-answer generation pattern. In contrast, when steering is applied, the top prediction becomes "Let", a common token used to begin step-by-step explanations. This qualitative observation suggests that steering shifts the model from an answer-generation trajectory to a reasoning trajectory at the earliest decoding step.

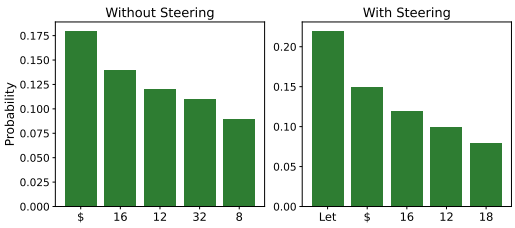


Figure 16: First-token probability distribution with and without steering.

A.10 Effect of Intervention Timing

We analyze how the effectiveness of steering depends on the timing of the intervention during generation. Following the feature ranking defined in Section 3.3, we consider both an intervention on the top-ranked feature (feature #8629 in LLaMA-3.1-8B-Instruct) and, for comparison, an intervention on the top-10 reasoning-related features. In all cases, we apply the intervention at a single decoding step while varying the step index.

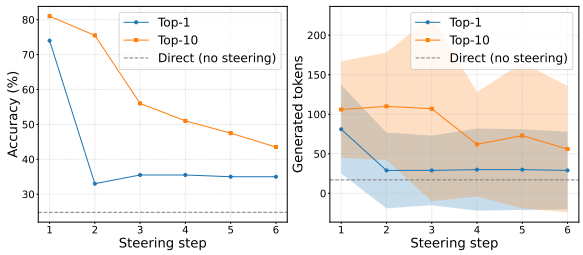


Figure 17: Effect of intervention timing when steering reasoning-related latent features on GSM8K with LLaMA-3.1-8B-Instruct. The figure reports accuracy (left) and generated token length (right) as a function of the decoding step at which the intervention is applied.

As shown in Figure 17, earlier interventions consistently yield stronger effects. Intuitively, entering a reasoning mode early gives the model more opportunity to carry out the intermediate computations needed for a correct solution, whereas late interventions may occur when the model is already close to, or has effectively committed to, an answer. This timing consideration is important for direct prompting, where some responses terminate within only a few tokens (e.g., $\sim$3 tokens), making later intervention steps ill-defined or ineffective.

Comparing top-1 and top-10 steering, both show a similar dependence on intervention timing, with earlier interventions being more effective, but their behaviors differ. As the intervention is delayed, top-10 steering weakens gradually, whereas top-1 steering drops sharply after the first step and quickly approaches the unsteered baseline. This suggests that a single feature can trigger reasoning when applied early, while sustained reasoning and its verbalization depend on multiple latent features, which is also reflected in the longer generations produced by top-10 steering.

A.11 Reasoning Feature as a Mode Indicator

We analyze whether the identified feature is associated with *entering a reasoning mode* or with *reasoning quality*. Specifically, we examine the correlation between feature activation and (i) prompting strategy (CoT vs. direct), and (ii) answer correctness within each strategy. Figure 5 shows a characteristic early activation pattern during answer generation. We quantify this pattern by comparing the maximum activation within the first 20 decoding steps under CoT and direct prompting.

As shown in Table 8, we find that the feature is activated significantly more strongly under CoT prompting than under direct prompting. Across 200 GSM8K samples, the activation of the feature is substantially higher for CoT than for direct prompting (point-biserial $r = 0.14$, $p = 0.006$), indicating a strong association with entering a reasoning-oriented generation mode. Computation details are provided in A.12.

Target Variable	corr. (p-value)	Mean (A)	Mean (B)
Reasoning Mode (A=CoT, B=Direct)	0.14 (0.006)	$0.10_{\pm 0.41}$	$0.01_{\pm 0.13}$
CoT Acc. (A=Correct, B=Wrong)	−0.02 (0.80)	$0.10_{\pm 0.40}$	$0.12_{\pm 0.43}$
Direct Acc. (A=Correct, B=Wrong)	−0.06 (0.40)	$0.00_{\pm 0.00}$	$0.02_{\pm 0.13}$

Table 8: Analysis of the correlation between feature activation values and different target variables. Activation values are collected from feature #8629 in LLaMA-3.1-8B-Instruct on GSM8K. We report the correlation coefficients (with p-values) and the mean feature scores corresponding to different groups in the target variable.

In contrast, when conditioning on a fixed prompting strategy (either CoT or direct), feature activation shows little correlation with answer correctness. As shown in Table 8, the feature does not reliably distinguish correct from incorrect answers under either strategy.

This interpretation also explains the limited gains observed when steering the feature under CoT prompting in Table 1: *once the model has already entered a reasoning mode, further amplifying the trigger alone does not substantially improve reasoning performance.*

A.12 Point-biserial Correlation Details

We measure the association between the activation magnitude of a latent feature and a binary target variable using the point-biserial correlation. For each sample i, let x_i denote the feature activation value (in our case, the maximum activation within the first 20 decoding steps), and let $y_i \in \{0, 1\}$ denote the binary label indicating whether the sample belongs to condition A ($y_i = 1$) or condition B ($y_i = 0$).

Let n_1 be the number of samples with $y_i = 1$ and n_0 the number of samples with $y_i = 0$, with $n = n_1 + n_0$. Define the group means

$$\bar{x}_1 = \frac{1}{n_1} \sum_{i:y_i=1} x_i, \qquad \bar{x}_0 = \frac{1}{n_0} \sum_{i:y_i=0} x_i,$$

and the overall sample standard deviation of x

$$s_x = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2}, \qquad \text{where } \bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i.$$

The point-biserial correlation coefficient is then

$$r_{pb} = \frac{\bar{x}_1 - \bar{x}_0}{s_x} \sqrt{\frac{n_1 n_0}{n^2}}.$$

(Equivalently, r_{pb} is identical to the Pearson correlation between x_i and the binary variable y_i when y_i is coded as $0/1$.)

To test significance under the null hypothesis $H_0 : r_{pb} = 0$, we use the standard t-test for correlation:

$$t = r_{pb} \sqrt{\frac{n-2}{1 - r_{pb}^2}},$$

with degrees of freedom $n-2$. We report two-sided p-values computed from the t distribution:

$$p = 2 \left(1 - F_{t(n-2)} \left(|t|\right)\right),$$

where $F_{t(n-2)}(\cdot)$ denotes the CDF of the t distribution with $n - 2$ degrees of freedom.

A.13 Ablation Study

We conduct a random feature ablation to verify that the observed steering effects are specific to the identified reasoning-related latent feature rather than arising from arbitrary perturbations. Using LLaMA-3.1-8B-Instruct under direct prompting, we compare steering the reasoning feature (#8629) with steering randomly selected non-zero SAE features from the same layer.

Feature	Acc. (%)	#Tok	Runs
#8629	73.3	83	-
Random	26.1±3.4	29±5	10

Table 9: Random feature ablation on GSM8K with LLaMA-3.1-8B-Instruct. Results for the reasoning-related feature are reported with a fixed random seed.

As shown in Table 9, steering the reasoning-related feature yields substantial gains in accuracy and induces longer generations, whereas random feature steering fails to produce comparable improvements. This result confirms that the steering effect is specific to the identified reasoning-related feature rather than a generic latent intervention.