Eliciting Chain-of-Thought in Base LLMs via Gradient-Based Representation Optimization

Zijian Wang[1], Yanxiang Ma[1], Chang Xu[1*]

[1]School of Computer Science, The University of Sydney
zwan0998@uni.sydney.edu.au, yama9404@uni.sydney.edu.au, c.xu@sydney.edu.au

Abstract

Chain-of-Thought (CoT) reasoning is a critical capability for large language models (LLMs), enabling them to tackle complex multi-step tasks. While base LLMs, pre-trained on general text corpora, often struggle with reasoning due to a lack of specialized training, recent studies reveal their latent reasoning potential tied to hidden states. However, existing hidden state manipulation methods, such as linear activation steering, suffer from limitations due to their rigid and unconstrained nature, often leading to distribution shifts and degraded text quality. In this work, we propose a novel approach for eliciting CoT reasoning from base LLMs through hidden state manipulation grounded in probabilistic conditional generation. By reformulating the challenge as an optimization problem with a balanced likelihood and prior regularization framework, our method guides hidden states toward reasoning-oriented trajectories while preserving linguistic coherence. Extensive evaluations across mathematical, commonsense, and logical reasoning benchmarks demonstrate that our approach consistently outperforms existing steering methods, offering a theoretically principled and effective solution for enhancing reasoning capabilities in base LLMs.

1 Introduction

Chain-of-Thought (CoT) reasoning is a vital capability for large language models (LLMs), enabling them to break down complex problems into intermediate steps for improved performance on complex tasks (Wei et al. 2022; Kojima et al. 2022). Models pre-trained solely on general text corpora, known as *Base LLMs*, often excel in fluency language generation but struggle with complex reasoning due to a lack of specialized training. In contrast, advanced models, termed *Reasoning LLMs*, incorporate reasoning-focused data (e.g., STEM, coding, synthetic examples) during pre-training and undergo extensive post-training, such as instruction fine-tuning and reinforcement learning, to enhance reasoning abilities (Guo et al. 2025; Yang et al. 2025).

Intriguingly, recent studies have uncovered latent reasoning potential within purely pre-trained Base LLMs. For instance, (Wang and Zhou 2024) demonstrates that, when sampling multiple responses from an LLM for a given problem, CoT-style answers are frequently present among the diverse outputs, even though the default greedy decoding yields only direct answers. Complementarily, (Wang and Xu 2025; Bi et al. 2025) shows that the presence of CoT can be effectively assessed from the model's hidden states, revealing a strong correlation between intrinsic reasoning capabilities and these internal representations. These findings underscore that pre-trained LLMs contain unlocked reasoning abilities tied to their hidden states, prompting efforts to enhance such models by directly manipulating these representations. Existing approaches primarily rely on linear activation steering, which computes and applies a pre-defined control vector with fixed steering strength in a single step to redirect the model's generation toward reasoning-oriented trajectories (Højer, Jarvis, and Heinrich 2025; Tang et al. 2025; Hong et al. 2025).

Despite its intuitive appeal and simplicity, linear activation steering suffers from significant limitations due to its inflexible framework. It fails to adapt to instance-specific variations and lacks a clear optimization objective to direct the steering process. Furthermore, it applies unbounded steering strength without well-defined constraints. As a result, this approach can disrupt the latent manifold structure, causing distribution shifts and degrading text quality, which ultimately leads to deviations from the model's natural language generation capabilities (Da Silva et al. 2025; von Rütte et al. 2024; Li et al. 2023a). Therefore, we pose the question: *Can we solely manipulate the hidden states to elicit CoT reasoning from a base LLM, without distorting its language quality?*

In this paper, we propose an alternative hidden states manipulation approach to elicit CoT reasoning from base LLMs based on probabilistic conditional generation principles. The foundation of our method lies in establishing a connection between hidden states manipulation and latent variable learning within a conditional text generation framework, which requires sampling appropriate hidden states from an intractable posterior distribution that satisfy the desired condition.

Inspired by Bayesian posterior estimation (Welling and Teh 2011), we reformulate this sampling challenge as an optimization problem comprising two essential components: a likelihood term and a prior term. The likelihood term, implemented as a pre-trained classifier, quantifies the probability that hidden states satisfy the target condition, thereby guiding these states toward the desired reasoning mode. Concurrently, the prior term functions as a regularization mechanism, con-

[*]Corresponding author

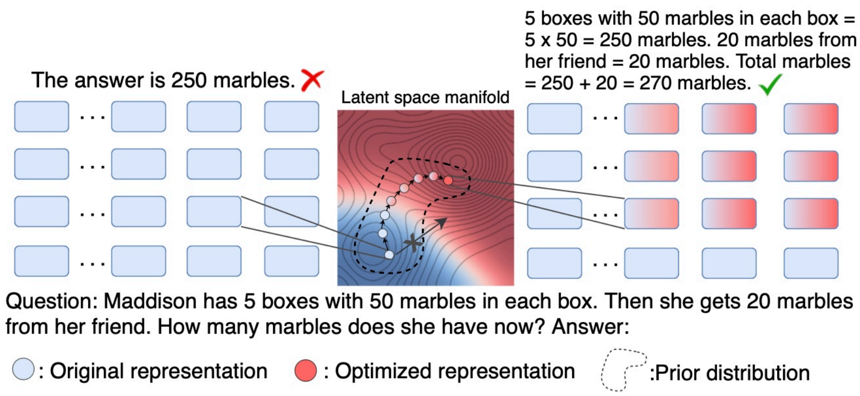

Figure 1: Our method optimizes hidden states within the base LLM's latent space to elicit Chain-of-Thought reasoning. Before optimization, the model produces direct answers without reasoning steps; after optimization, it generates chain of thought content. Unlike linear activation steering that may disrupt the language prior without regulations, our approach implements guided optimization with principled constraints.

straining hidden states to remain proximate to the original manifold structure, thus preserving linguistic coherence in the generated text. These complementary objectives are carefully balanced to ensure optimization convergence while maintaining both reasoning effectiveness and linguistic fluency. During LLM inference, our method employs gradient-based optimization to compute hidden states that maximize this composite objective function. This approach effectively navigates the representation space to satisfy the CoT condition while the prior regularization avoids the distribution shift and degradation of text quality.

We evaluate our method on several reasoning benchmarks, including mathematical, commonsense, and logical reasoning tasks. Our method is proven to be effective in eliciting CoT reasoning capabilities in base LLMs compared to vector arithmetic methods on multiple major benchmarks.

The main contributions of this work are:

1: We propose a new hidden states manipulation method based on probabilistic conditional generation theory to elicit CoT reasoning from base LLMs. Our approach is theoretically grounded and principled, offering advantages over conventional rigid vector arithmetic methods.

2: We develop a comprehensive optimization framework that balances likelihood maximization with prior regularization, effectively guiding hidden states toward reasoning capabilities while preserving representational integrity. We establish theoretical bounds to guarantee this trade-off.

3: We demonstrate our method's effectiveness through extensive evaluation across diverse reasoning benchmarks, consistently outperforming existing hidden states steering approaches.

2 Background

2.1 LLM Text Generation

LLMs generate text in an autoregressive manner. Given a sequence of tokens $X = (x_1, x_2, \cdots, x_n)$ with length n, the probability of the sequence $p(X)$ is factorized as:

$$p(X) = \prod_{i=1}^{n} p(x_i | x_0, \cdots, x_{i-1}). \tag{1}$$

In transformer architectures, the contextual information from previous tokens is encoded into hidden states through multiple layers of self-attention and feed-forward networks. For a token at position i, the hidden state h_i captures the representation of the sequence processed so far $h_i = f_{\text{transformer}}(x_{<i})$. where $f_{\text{transformer}}$ represents the transformation through the model's layers, and $x_{<i} = (x_0, x_1, \ldots, x_{i-1})$ is the input context. We omit the layer index of h_i for simplicity.

The distribution for each token $p(x_i | x_0, \cdots, x_{i-1})$ can then be expressed by the hidden states as

$$p(x_i | x_0, \cdots, x_{i-1}) = p(x_i | h_i) = \text{softmax}(h_i W_o), \tag{2}$$

where W_o is the weight matrix of the output head. This formulation highlights that next-token prediction is fundamentally determined by the hidden representation h_i, which encodes necessary contextual information for generation.

2.2 Conditional Text Generation

Conditional generation aims to produce text X that satisfies a specific condition c, formally expressed as sampling from the distribution $p(X|c)$, which is a wide theoretical paradigm in language generation, such as attribute transfer (Wang, Hua, and Wan 2019) and controllable text generation (Dathathri et al. 2019), where the condition c is a high level semantic attribute. Due to the nature of LLM forward computation, the text and hidden states are jointly defined in a distribution. Formally, we can formulate the conditional generation process as

$$p(X, h|c) = p(X|h, c)p(h|c) = p(X|h)p(h|c). \tag{3}$$

The equality $p(X|h, c) = p(X|h)$ holds because once the hidden state h is fixed, the LLM's decoding process is fully determined and independent of the condition c. This formulation reveals that h functions as a mediator between the condition c and the generated text X.

To generate text satisfying condition c, we can employ ancestral sampling along the causal path $c \to h \to X$. This involves two steps, first sampling a hidden state from $p(h|c)$ that meets the desired condition, then generating text from $p(X|h)$ via the decoding process of LLMs. While the LLMs architecture already parameterized the second distribution $p(X|h)$, the key challenge lies in efficiently sampling from $p(h|c)$. The sampling process is normally intractable due to the high dimensionality of hidden states and the complex condition-representation relationship.

3 Methodology

In this section, we present our approach to elicit chain-of-thought reasoning in base LLMs through hidden states optimization. We first formulate the problem within a Bayesian posterior inference framework, then derive an analytical optimization objective form with two complementary targets, and develop a gradient-based optimization method for its practical implementation.

3.1 Problem Formulation

In the conditional generation framework the hidden state h serves as the causal mediator between CoT condition c and generated text X. By building this framework, we confront the fundamental challenge of sampling from the posterior distribution $p(h|c)$ to obtain hidden states that satisfy our desired condition. Rather than attempting sampling from the posterior distribution, we employ Maximum Posterior (MAP) estimation (Welling and Teh 2011) to identify the mode of the posterior distribution, formally,

$$h^* = \text{argmax}_h[\log p(h|c)]. \tag{4}$$

Practically, we apply Bayesian rule to decompose the posterior distribution as $p(h|c) = \frac{p(c|h)p(h)}{p(c)} \propto p(c|h)p(h)$ and the optimization target in Eq. 4is equivalent to:

$$h^* = \text{argmax}_h[\log p(c|h) + \log p(h)] \tag{5}$$

where $p(c|h)$ is the likelihood and $p(h)$ is the prior term. To solve this optimization in practice, we use gradient-based methods starting from h_0, the original hidden state obtained from the LLM's initial forward pass on the input context.

3.2 Deriving the Optimization Objective

We first introduce how we model the prior and likelihood term to derive a analytical solution for the optimization target. And then, we revise the optimization process with an adaptive term and a regularization noise. Finally, we demonstrate the entire process of our method.

Modeling Prior In the context of posterior estimation, the prior distribution $p(h)$ serves to constrain the optimization process by penalizing large deviations from the original hidden state h_0 (Fortuin 2022), ensuring that the resulting hidden state remains plausible and stable within a reasonable neighborhood. We model this prior as $p(h) \propto \exp(-d(h, h_0))$, where the probability density decreases exponentially with the L2 distance $d(h, h_0)$ from h_0.

$$-\log p(h) \approx d(h, h_0) \propto \frac{1}{2}\|h - h_0\|^2 \tag{6}$$

More theoretical justification for this L2 regularization term is provided in the appendix.

Likelihood Term For the likelihood term $p(c|h)$, we implement a Multi-Layer Perceptron classifier $f_\theta(h)$ estimating the probability that hidden state h exhibits the target condition c. Trained on a contrastive dataset of hidden states from both CoT and non-CoT responses (labeled positive and negative, respectively), the classifier is optimized using cross-entropy loss to effectively distinguish between reasoning and non-reasoning hidden states. More details are provided in the experiments.

With the above two components, we can rewrite the optimization target as:

$$h^* = \text{argmax}_h[\log f_\theta(h) - d(h, h_0)] \tag{7}$$

And we can use gradient ascent to update the hidden state with a step size α:

$$h_{t+1} = h_t + \alpha \nabla_h[\log f_\theta(h_t) - d(h_t, h_0)] \tag{8}$$

Adaptive Step Size and Random Noise Drawing from gradient descent principles (Ge et al. 2019), we implement an adaptive step size α_t that dynamically decays as the $f_\theta(h_t)$ approaches the target value, ensuring both efficient convergence and optimization stability, defined as $\alpha_t = \alpha_0 \cdot \frac{|\tau - f(h_t)|}{1+f(h_0)}$ where τ is the target value of the optimization and α_0 is the initial step size. Further, to facilitate exploration beyond local minima and ensure authentic sampling from the distribution (Welling and Teh 2011), we incorporate Gaussian noise $z \sim \mathcal{N}(0, I)$ into the optimization process.

Using gradient ascent with the adaptive step length and random noise, the iterative updating on h at timestep t can finally be formulated as

$$h_{t+1} = h_t + \alpha_t \nabla_h[\log f_\theta(h_t) - \lambda d(h_t, h_0)] + \sqrt{\alpha}_t \cdot z \tag{9}$$

where λ is a hyperparameter that controls the trade-off between the likelihood term and the prior distribution. The trade-off between them helps to balance two competing objectives: steering the hidden state toward CoT reasoning mode while maintaining its proximity to the original language manifold. The complete optimization procedure is presented in Algorithm 1.

3.3 Balancing Likelihood and Prior

Determining proper bounds for λ is crucial, as inappropriate values may fail to reach the CoT mode or push hidden states off-manifold, resulting in incoherent text generation. In this section, we establish theoretical bounds for λ from two perspectives: ensuring gradient alignment with the likelihood objective, and maintaining appropriate step magnitude during update.

Aligning Optimization Direction The gradient of the likelihood term $\nabla_h(\log f_\theta(h_t))$ directly points toward the CoT mode. For effective optimization, we expect the overall gradient of Eq. 9 to closely align with this CoT-inducing direction, which we achieve by maximizing their cosine similarity, ensuring the updates predominantly towards CoT mode. Denote the gradient of h in Eq. 9 is ∇^{**}, and denote $\nabla((\log f_\theta(h_t)))$ is ∇^*. By maximizing the cosine similarity $Cos(\nabla^*, \nabla^{**})$, we can derive the upper bound of λ.

Lemma 1. *Assume that the cosine similarity is larger than $1 - \epsilon_c$, where ϵ_c is a positive small number close to 0. At any timestep t, λ must be smaller than the upper bound as*

$$\lambda \leqslant \frac{\epsilon_c \sum_n^{i=1} \nabla_i^{*2}}{2 \sum_n^{i=1} h_{ti} \times \nabla_i^*} \tag{10}$$

where n is the length of h_t, and h_{ti} and ∇_i^* denotes the i-th item in h_t and ∇^*, respectively.

The detailed proof of Lemma 1 is in the appendix. With the upper bound stands, λ can be guaranteed to be a very small positive number. This helps keep each step of the gradient up on the correct path.

Maximizing the Distance. Beyond proper directional alignment, we also need to calibrate step magnitude during optimization to prevent drifting outside from the distribution. Therefore, the distance between $d(h_{t+1}, h_0)$ is expected to get smaller at each iteration. At timestep t, we assume that $\hat{h}_{t+1}$ is a special case of h_{t+1} where $\lambda = 0$ as

$$\hat{h}_{t+1} = h_t + \alpha_t \nabla_h (\log f_\theta(h_t)) + \sqrt{\alpha_t} \cdot z. \tag{11}$$

We can minimize $d(h_{t+1}, h_0)$ by maximizing $d(\hat{h}_{t+1}, h_0) - d(h_{t+1}, h_0)$. Detailed proof is in the appendix.

Lemma 2. *Assume that $\|\hat{h}_{t+1}\| - \|h_{t+1}\|$ is larger than a positive number ϵ_d. Ignoring the regularization term and denoting the cosine of angle $< h_{t+1}, \nabla_h(\|h_t\|) >$ as C_t, the lower bound of λ can then be derived using the cosine theorem as*

$$\lambda \geqslant \frac{\nabla^* + \|h_t\| \cdot C_t - \sqrt{[\nabla^* + \|h_t\| \cdot C_t]^2 - \epsilon_t}}{\nabla_h(\|h_t\|)} \tag{12}$$

See detailed proof in the appendix.

With Lemma 1 and Lemma 2 working together, the range of λ is

$$\frac{\nabla^* + \|h_t\| \cdot C_t - \sqrt{[\nabla^* + \|h_t\| \cdot C_t]^2 - \epsilon_t}}{\nabla_h(\|h_t\|)} \leqslant \lambda \leqslant \frac{\epsilon_c \sum_n^{i=1} \nabla_i^{*2}}{2 \sum_n^{i=1} h_{ti} \times \nabla_i^*} \tag{13}$$

With this constraint, the optimization maintains both update directional alignment and appropriate step size magnitude. Each optimization step ensures the hidden state moves efficiently toward the CoT reasoning mode while remaining within the proximity of the natural language distribution.

Algorithm 1: Hidden State Optimization for Chain-of-Thought Reasoning

Require: A base LLM, trained classifier f_θ, input tokens X, target layer l, trade-off λ, step size α_0, threshold τ, maximum iterations T, generation length n
Ensure: Hidden stats are optimized to desired mode.
for $i = 1$ to n **do**
  $h_0^{(l)} \leftarrow$ LLM.forward(X, l) ▷ Layer l Forward
  if $f_\theta(h^{(l)}) < 0.5$ **then** ▷ Optimize negative samples
    $h_t^{(l)} \leftarrow h_0^{(l)}$ ▷ Initialize optimization
    for $t = 1$ to T **do**
      $g \leftarrow \nabla_h[\log f_\theta(h_t^{(l)}) - \lambda d(h_t^{(l)}, h_0^{(l)})]$ ▷ Compute gradient
      $\alpha_t \leftarrow \alpha_0 \cdot \frac{|\tau - f_\theta(h_t^{(l)})|}{|f_\theta(h_0^{(l)})| + \epsilon}$ ▷ Adaptive step length
      $h_{t+1}^{(l)} \leftarrow h_t^{(l)} + \alpha_t g + \sqrt{\alpha_t} \cdot z$ ▷ Update
      if $f_\theta(h_{t+1}^{(l)}) > \tau$ **then**
        break ▷ Stop when condition is satisfied
      end if
    end for
  end if
  LLM.update_layer_activation$(l, h_{t+1}^{(l)})$
  $x_i \leftarrow$ LLM.forward_from_layer(l) ▷ Update hidden state and continue forward pass
  $X \leftarrow X \cup x_i$ ▷ Update input context
end for
return X ▷ Return generated text

4 Experiments

4.1 Setup

Datasets: We evaluate our method on three reasoning task types: math, commonsense, and logical. For math, we select four well-known datasets: GSM8K (Cobbe et al. 2021), MultiArith (Roy and Roth 2016), SVAMP (Patel, Bhattamishra, and Goyal 2021), and MAWPS (Koncel-Kedziorski et al. 2016), which encompass a range of problem difficulties and structures. We also choose two popular commonsense datasets: CommonsenseQA(C-QA) (Talmor et al. 2018) and StrategyQA(S-QA) (Geva et al. 2021) and two popular logical datasets: Objects Tracking (Srivastava et al. 2022) and Coin Flip (Srivastava et al. 2022). **LLMs:** We choose four solely pre-trained base LLMs: Mistral-7b (Jiang et al. 2023), Gemma-7b (Team et al. 2024), Qwen1.5-4b (Bai et al. 2023) and Phi-1.5 (Li et al. 2023b), running on NVIDIA 4090 GPUs.

Classifier and Hidden states: To build the training dataset for our classifier, we first generate paired responses for each problem. We input problems from the GSM8K training set into Mistral-7b and sample multiple responses per problem. These responses are then classified by GPT-4 as either CoT (with explicit reasoning steps and correct solutions) or non-CoT (with direct answers or irrelevant content), resulting in a dataset of 2000 response pairs. This dataset is split into training and validation sets in an 8:2 ratio. This paired dataset is used to train classifiers for four base LLMs.

Next, to obtain the hidden states for training the classifiers, we concatenate the questions and responses and feed them into the LLM. At each layer of the LLM, we extract the hidden states corresponding to CoT and non-CoT responses. These hidden states are then used to train a classifier specific to each layer. Our classifier f_θ is designed as a two-layer MLP with ReLU activations and a sigmoid output. It is trained on these contrastive hidden states using cross-entropy loss to distinguish between CoT and non-CoT representations effectively. More classifier details are in the appendix.

Specifically, we consider three types of hidden represen-

LLM	Method	Math				Commonsense and Logic			
		GSM8K	MultiArith	SVAMP	MAWPS	C-QA	S-QA	Obj-T	C-Flip
Mistral-7b	Vanilla	11.32%	15.55%	52.66%	56.61%	56.06%	50.59%	33.08%	48.75%
	C-DIM	13.42%	17.86%	54.76%	57.26%	55.47%	50.31%	32.84%	49.65%
	C-PCA	14.52%	17.65%	55.24%	57.87%	56.32%	51.85%	33.83%	48.49%
	C-LR	15.86%	18.93%	56.97%	58.97%	56.94%	49.63%	34.05%	49.75%
	P-SVM	15.74%	20.86%	56.48%	**59.45%**	55.83%	51.85%	35.64%	50.74%
	DA	15.15%	19.35%	55.14%	57.63%	54.21%	50.04%	32.45%	49.35%
	Ours	**17.24%**	**23.33%**	**57.33%**	59.20%	**57.19%**	**52.04%**	**34.57%**	**50.87%**
Gemma-7b	Vanilla	21.62%	30.55%	53.33%	64.44%	52.92%	42.83%	31.68%	45.54%
	C-DIM	23.64%	35.63%	52.62%	65.58%	51.38%	43.54%	32.84%	46.75%
	C-PCA	24.72%	39.48%	55.34%	65.74%	52.74%	43.79%	33.72%	46.84%
	C-LR	26.76%	39.74%	55.25%	67.86%	53.41%	42.86%	32.31%	47.71%
	P-SVM	26.32%	40.61%	57.83%	67.53%	53.56%	43.61%	32.86%	46.53%
	DA	25.75%	39.43%	56.47%	66.49%	52.75%	42.35%	31.26%	47.52%
	Ours	**28.79%**	**45.11%**	**59.67%**	**69.92%**	**54.78%**	**44.42%**	**34.54%**	**48.81%**
Qwen1.5-4b	Vanilla	49.26%	89.00%	54.33%	65.24%	71.67%	53.41%	28.54%	52.78%
	C-DIM	50.63%	88.74%	55.14%	64.62%	72.74%	54.52%	30.39%	53.41%
	C-PCA	49.74%	89.78%	55.45%	65.04%	71.09%	55.75%	31.42%	52.63%
	C-LR	50.94%	90.32%	55.87%	66.13%	72.74%	55.93%	30.84%	53.82%
	P-SVM	51.24%	90.15%	56.24%	67.45%	73.12%	57.34%	30.75%	54.84%
	DA	50.02%	**91.73%**	55.39%	66.56%	71.42%	55.65%	30.46%	53.65%
	Ours	**51.33%**	91.66%	**57.33%**	**68.62%**	**73.48%**	**59.52%**	**32.45%**	**55.63%**
Phi-1.5	Vanilla	5.69%	7.78%	11.67%	15.29%	27.93%	37.53%	31.60%	49.34%
	C-DIM	5.43%	10.45%	13.55%	15.84%	28.14%	37.24%	31.75%	49.72%
	C-PCA	5.84%	11.53%	12.64%	16.94%	29.31%	37.85%	32.48%	50.04%
	C-LR	6.47%	13.75%	13.93%	16.08%	28.75%	38.14%	31.04%	50.75%
	P-SVM	6.24%	13.54%	14.73%	16.75%	28.63%	37.95%	32.54%	49.47%
	DA	**6.85%**	14.56%	14.91%	16.31%	29.18%	38.63%	32.79%	50.15%
	Ours	6.74%	**16.67%**	**15.00%**	**17.50%**	**29.84%**	**39.74%**	**34.33%**	**51.27%**

Table 1: Problem-solving accuracy comparison between our method and baseline approaches across diverse reasoning tasks. Bold values indicate best performance for each model-dataset combination. Our approach consistently improves performance across nearly all models and datasets. Vanilla is defaultgreedy decoding performance.

tations in each transformer layer: the activations from the self-attention component(ATTN), the multi-layer perceptron(MLP) component, and the integrated layer output(INT-Layer). During optimization, we select the hidden state type with the best classification performance to ensure effective steering of the model's reasoning capabilities. Additionally, we choose the optimization layers from the top 50% of layers based on their classification performance. The justification for this selection strategy is detailed in Section 4.5.

4.2 Evaluation Metrics

Answer Accuracy: The correctness of answers, which tends to be higher when CoT reasoning is present in the generated text. **Fluency**: Weighted average of bi- and tri-gram entropies, calculated as $-\sum_k f(k)\log_2 f(k)$, where $f(\cdot)$ is the n-gram frequency and k denotes each unique n-gram in the text (Meng et al. 2022). **Perplexity**: Measure the model's ability to model the response, with higher values indicating weaker modeling capability and a shift in distribution during optimization. **GPT4o Judge Score**: We use GPT-4o to evaluate the reasoning content in generated text, scoring from 0 (direct answers without reasoning) to 1 (detailed chain-of-thought content). The prompt is the appendix.

4.3 Answer Accuracy Results

Config: We set hyperparameters as follows: maximum iterations at 200, α_0 at 0.1, λ at 0.01, and τ at 0.9. Hyperparameter analysis is discussed in Section 4.4. For input formatting, we use a straightforward question-answer template: "Question:[question]\nAnswer:" without additional prompting techniques. For hidden states, we select integrated layer output for Mistral-7B and Gemma-7B, and self-attention activations for Qwen1.5-4b and Phi-1.5, based on their classification performance.

Baselines: We compare action steering methods that modify hidden states with control vectors from various techniques:

- Difference in Mean(C-DiM) (Højer, Jarvis, and Heinrich 2025): Constructs a control vector by calculating the difference between average representations of CoT and non-CoT examples.

- Principal Component Analysis(C-PCA) (Zou et al. 2023): Creates a control vector by finding the principal component of differences between sampled CoT and non-CoT representations.

- Logistic Regression(C-LR) (von Rütte et al. 2024): Trains a logistic regression model on CoT and non-CoT hidden states, using the weight vector as the control vector for reasoning direction.

We also compare activation editing methods:

- Boundary Projection in SVM(P-SVM) (Huang, Chen, and Umrawal 2025): Trains an SVM to obtain a classification hyperplane, then projects negative sample representations onto this boundary surface.

- Directional Ablation(DA) (Arditi et al. 2024): Ablates the negative DiM control vector by projecting activations

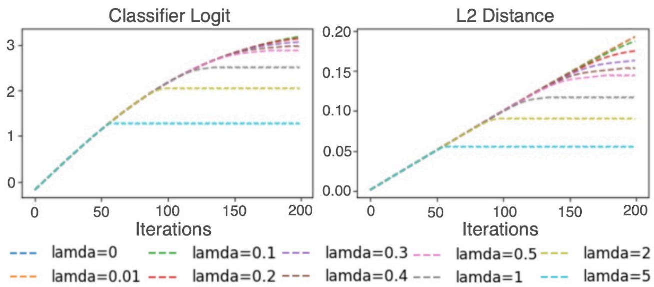

Figure 2: Impact of different λ values on the iterative optimization process, demonstrating how this hyperparameter influences the optimization convergence.

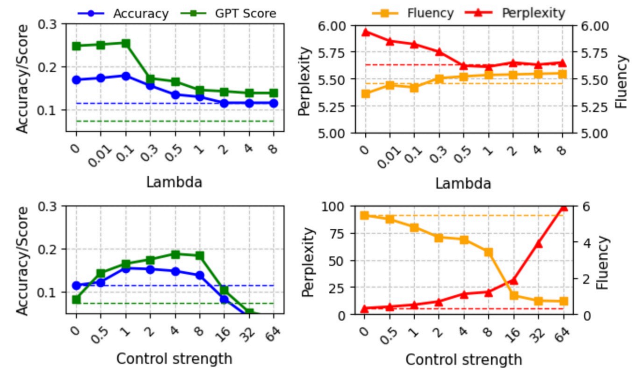

Figure 3: Upper: Impact of different λ values on our method's performance, roughly showing the optimal range. Lower: Comparative analysis of activation steering with different strength. The dashed line in the figure represents the performance of the greedy decoding baseline.

orthogonal to this direction, eliminating the model's tendency to refrain from reasoning.

All baseline methods intervene at the same layers as our method, and the strength of the control vectors is uniformly set to 1 for consistency in comparison. For all experiments, the sampling strategy in text generation is set to greedy sampling.

Analysis: Table 1 presents a comprehensive comparison of answer accuracy between our method and various baselines across multiple LLMs and tasks. The Vanilla baseline represents the performance of unmodified greedy sampling without any intervention on the model's hidden states. Control vector-based methods achieve only marginal improvements in almost all tasks, likely due to their inability to precisely steer activations toward optimal regions and their lack of regularization during the steering process.

Our method, in contrast, demonstrates substantial improvements across nearly all LLMs and domains, with particularly impressive results on math reasoning benchmarks—achieving performance gains of up to 14.56% on MultiArith using Gemma-7b and 8.89% on MultiArith using Phi-1.5. These significant enhancements validate our approach's effectiveness in unlocking latent chain-of-thought reasoning capabilities inherent in base models.

While improvements on commonsense and logical reasoning tasks are more modest, they remain consistent, with an average gain of 1.8% over other baselines. This performance differential suggests these tasks may demand more domain-specific knowledge beyond pure reasoning ability. Furthermore, all classifiers are trained on Mistral-7B data but generalize well to other models, showing strong method transferability.

4.4 Optimization Config Analysis

In this section, we analyze the sensitivity to hyperparameters like λ and target threshold τ. λ balances likelihood and prior terms and τ sets the classifier output threshold.

Trade-off between likelihood and prior terms: Figure 2 illustrates how varying λ affects the optimization dynamics through three key metrics: logit value of classifier $f_\theta(\tilde{h})$, L2 distance $|h_t - h_0|^2$ across λ values from 0 to 5.

When λ is small (0-0.1), the likelihood term dominates the optimization, leading to a consistent increase in $f_\theta(h_t)$, with values of $f_\theta(h_t)$ being very close across different λ in this range, while the L2 distance grows as h_t diverges from the initial state h_0, similarly showing close values among different λ. As λ exceeds 0.1, the prior term becomes more influential, leading to quicker saturation of classifier output at lower values. Meanwhile, the L2 distance converges faster to smaller final values, accelerating the optimization process to earlier equilibrium.

Figure 3 illustrates how our method's performance changes as the hyperparameter λ varies. For λ in the range of 0-0.01, we observe an increase in performance alongside decreasing language quality, indicating an optimal operating region where the optimization process effectively balances the trade-off in Section 3.3. This sweet spot enables enhanced reasoning capabilities while maintaining high text quality. As λ exceeds 0.01, language quality drops significantly, and reasoning performance also decline, indicating the prior term dominates, restricting exploration of effective reasoning paths. Nevertheless, across all λ values, both metrics consistently surpass the vanilla baseline, showing enhanced reasoning capabilities regardless of hyperparameter settings.

For comparison, we plot the curves for C-LR, which uses an additive control vector with adjustable strength. Results show that C-LR's reasoning performance improves and language quality decreases for strength values between 0 and 2, yet it still underperforms compared to our method even at optimal strength. Moreover, as strength increases, performance collapses below the greedy decoding baseline, and language quality deteriorates drastically, severely disrupting the model's output structure. This highlights our method's superior stability and effectiveness across its parameter range.

Evaluation of τ: In this evaluation, we assess the sensitivity of our method to τ, the target threshold for classifier output, which is crucial for guiding the optimization process.

Table 2 show the results of varying τ from 0.5 to 0.99 to observe its impact, using Mistral-7b on the GSM8K dataset for validation. Here, $\tau = 0.5$ represents the classification boundary. As τ increases from 0.5, reasoning performance, reflected in both accuracy and GPT score, steadily improves,

τ	Accuracy	GPT-Score	Fluency	Perplexity
0.50	13.57%	0.16	5.56	5.32
0.60	14.56%	0.18	5.46	5.42
0.70	15.56%	0.19	5.35	5.47
0.80	16.56%	0.23	5.33	5.53
0.90	18.24%	0.25	5.27	5.59
0.92	18.26%	0.26	5.25	5.61
0.95	18.35%	0.26	5.25	5.63
0.99	18.45%	0.27	5.24	5.65

Table 2: Analysis of different τ values on the iterative optimization process, using Mistral-7b on GSM8K

LLM	layers-Topk	MLP(81.23)	ATTN(83.45)	Int-Layer(91.62)
Mistral-7b	10%	12.64%	13.42%	15.86%
	20%	14.52%	14.74%	16.83%
	30%	15.24%	15.51%	17.86%
	40%	15.74%	16.26%	18.93%
	50%	15.74%	17.86%	18.24%
	70%	15.74%	17.86%	18.56%
	90%	16.83%	18.93%	18.78%
LLM	layers-Topk	MLP(83.47)	ATTN(81.73)	Int-Layer(92.31)
Gemma-7b	10%	21.79%	21.45%	23.86%
	20%	22.37%	22.26%	24.73%
	30%	23.65%	23.59%	25.93%
	40%	24.34%	24.07%	27.46%
	50%	24.84%	24.85%	28.79%
	70%	24.98%	25.14%	29.14%
	90%	25.14%	25.55%	29.32%

Table 3: Optimization layers and hidden states analysis, using Mistral-7b and Gemma-7b in GSM8K. We select the top-k layers based on the classification performance of the MLP, ATTN, and Int-Layer hidden states. The average classification F1 scores for each hidden state type are annotated.

reaching saturation around $\tau = 0.9$, while language quality remains stable. When τ exceeds 0.9, the performance gains become marginal, indicating that further increases in the threshold beyond this point yield diminishing returns in reasoning improvement.

4.5 Hidden States and Layer Analysis

In this experiment, we examine the effect of varying hidden states and layers during representation optimization.

Table 3 displays the answer accuracy for Mistral-7b and Gemma-7b across these variations, along with the classification F1 score for each hidden state. Layers are ranked by classification accuracy in descending order, and the top-k layers are chosen for optimization.

The results show a correlation between classification performance and optimization effectiveness. Both models achieve the greatest improvements with hidden states yielding the best classification results. As the proportion of top-k layers rises from 10% to 50%, accuracy improves significantly, but beyond 50%, gains become marginal, making additional layers less cost-effective due to increased computational demands. More comprehensive overhead analysis is provided in the appendix.

5 Related Work

Representation Engineering in LLMs Representation engineering (Zou et al. 2023) posits that LLM hidden states encode high-level concepts as causal factors, enabling output control through manipulation of these representations. This approach has proven effective across domains: reducing hallucinations through truthfulness patterns (Li et al. 2023a; Xiao et al. 2024; Liu, Ye, and Zou 2024), rejecting harmful content via activation patterns (Lee et al. 2024; Wang, Whyte, and Xu 2024), modulating pesonality traits and political perspectives (Zhu et al. 2024; Kim, Evans, and Schein 2025), and enhancing reasoning capabilities (Højer, Jarvis, and Heinrich 2025; Wang and Xu 2025; Hong et al. 2025; Tang et al. 2025; **?**). Despite these advances, most are remain constrained by linear steering paradigms without sophisticated optimization frameworks.

Reasoning Ability Enhancement in LLMs Methods to improve LLMs' reasoning abilities can be categorized into tuning-based and prompting-based approaches. Tuning-based methods rely on high-quality data and supervision. (Zelikman et al. 2022; Zhang et al. 2024; Hoffman et al. 2024) bootstrap reasoning via iterative generation and filtering; Deepseek-R1 (Guo et al. 2025) employs outcome-based rewards to reinforce reasoning capabilities.Prompting-based methods utilize LLMs' few-shot learning (Wei et al. 2022) and instruction-following abilities (Yao et al. 2023; Kojima et al. 2022) to elicit reasoning patterns through carefully designed prompts. While effective, these approaches rely on external resources to enhance reasoning, rather than leveraging the intrinsic reasoning capabilities already encoded within LLMs' representations.

6 Discussion and Limitations

While effective, as an inference-time computation technique, our method can only unlock intrinsic reasoning abilities rather than inject new capabilities. For models with limited intrinsic reasoning capacity established during pre-training, our approach cannot create non-existent capabilities. Additionally, our method's effectiveness depends significantly on the classification performance related to representations and layer selection; improper choices may lead to suboptimal results by failing to accurately capture the relevant reasoning states. Looking ahead, we envision integrating our method with post-training techniques to further amplify the reasoning capabilities of LLMs.

7 Conclusion

We present a principled gradient-based activation optimization method to elicit CoT reasoning from base LLMs. Grounded in probabilistic generation theory, our approach derives an analytical objective from Bayesian principles that balances likelihood and prior terms—theoretically guaranteed to activate reasoning without causing distribution shift. Experiments across diverse reasoning benchmarks demonstrate our method consistently outperforms vector arithmetic approaches while maintaining textual coherence. This work offers a theoretically sound approach to enhance reasoning in foundation models without additional training.

References

Arditi, A.; Obeso, O.; Syed, A.; Paleka, D.; Panickssery, N.; Gurnee, W.; and Nanda, N. 2024. Refusal in language models is mediated by a single direction. *Advances in Neural Information Processing Systems*, 37: 136037–136083.

Bai, J.; Bai, S.; Chu, Y.; Cui, Z.; Dang, K.; Deng, X.; Fan, Y.; Ge, W.; Han, Y.; Huang, F.; et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Bi, J.; Yan, D.; Wang, Y.; Huang, W.; Chen, H.; Wan, G.; Ye, M.; Xiao, X.; Schuetze, H.; Tresp, V.; et al. 2025. Cot-kinetics: A theoretical modeling assessing lrm reasoning process. *arXiv preprint arXiv:2505.13408*.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. 2021. Training verifiers to solve math word problems, 2021. *URL https://arxiv. org/abs/2110.14168*.

Da Silva, P. Q.; Sethuraman, H.; Rajagopal, D.; Hajishirzi, H.; and Kumar, S. 2025. Steering off Course: Reliability Challenges in Steering Language Models. *arXiv preprint arXiv:2504.04635*.

Dathathri, S.; Madotto, A.; Lan, J.; Hung, J.; Frank, E.; Molino, P.; Yosinski, J.; and Liu, R. 2019. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*.

Fortuin, V. 2022. Priors in bayesian deep learning: A review. *International Statistical Review*, 90(3): 563–591.

Ge, R.; Kakade, S. M.; Kidambi, R.; and Netrapalli, P. 2019. *The step decay schedule: a near optimal, geometrically decaying learning rate procedure for least squares*. Red Hook, NY, USA: Curran Associates Inc.

Geva, M.; Khashabi, D.; Segal, E.; Khot, T.; Roth, D.; and Berant, J. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9: 346–361.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Hoffman, M. D.; Phan, D.; Dohan, D.; Douglas, S.; Le, T. A.; Parisi, A.; Sountsov, P.; Sutton, C.; Vikram, S.; and A Saurous, R. 2024. Training chain-of-thought via latent-variable inference. *Advances in Neural Information Processing Systems*, 36.

Højer, B.; Jarvis, O.; and Heinrich, S. 2025. Improving Reasoning Performance in Large Language Models via Representation Engineering. *arXiv preprint arXiv:2504.19483*.

Hong, Y.; Zhou, D.; Cao, M.; Yu, L.; and Jin, Z. 2025. The Reasoning-Memorization Interplay in Language Models Is Mediated by a Single Direction. *arXiv preprint arXiv:2503.23084*.

Huang, Y.; Chen, D.; and Umrawal, A. K. 2025. JAM: Controllable and Responsible Text Generation via Causal Reasoning and Latent Vector Manipulation. *arXiv preprint arXiv:2502.20684*.

Jiang, A. Q.; Sablayrolles, A.; Mensch, A.; Bamford, C.; Chaplot, D. S.; Casas, D. d. l.; Bressand, F.; Lengyel, G.; Lample, G.; Saulnier, L.; et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825*.

Kim, J.; Evans, J.; and Schein, A. 2025. Linear Representations of Political Perspective Emerge in Large Language Models. *arXiv preprint arXiv:2503.02080*.

Kojima, T.; Gu, S. S.; Reid, M.; Matsuo, Y.; and Iwasawa, Y. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35: 22199–22213.

Koncel-Kedziorski, R.; Roy, S.; Amini, A.; Kushman, N.; and Hajishirzi, H. 2016. MAWPS: A math word problem repository. In *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: human language technologies*, 1152–1157.

Lee, B. W.; Padhi, I.; Ramamurthy, K. N.; Miehling, E.; Dognin, P.; Nagireddy, M.; and Dhurandhar, A. 2024. Programming refusal with conditional activation steering. *arXiv preprint arXiv:2409.05907*.

Li, K.; Patel, O.; Viégas, F.; Pfister, H.; and Wattenberg, M. 2023a. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36: 41451–41530.

Li, Y.; Bubeck, S.; Eldan, R.; Del Giorno, A.; Gunasekar, S.; and Lee, Y. T. 2023b. Textbooks are all you need ii: phi-1.5 technical report. *arXiv preprint arXiv:2309.05463*.

Liu, S.; Ye, H.; and Zou, J. 2024. Reducing hallucinations in large vision-language models via latent space steering. In *The Thirteenth International Conference on Learning Representations*.

Meng, K.; Sharma, A. S.; Andonian, A.; Belinkov, Y.; and Bau, D. 2022. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*.

Patel, A.; Bhattamishra, S.; and Goyal, N. 2021. Are NLP models really able to solve simple math word problems? *arXiv preprint arXiv:2103.07191*.

Roy, S.; and Roth, D. 2016. Solving general arithmetic word problems. *arXiv preprint arXiv:1608.01413*.

Srivastava, A.; Rastogi, A.; Rao, A.; Shoeb, A. A. M.; Abid, A.; Fisch, A.; Brown, A. R.; Santoro, A.; Gupta, A.; Garriga-Alonso, A.; et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Talmor, A.; Herzig, J.; Lourie, N.; and Berant, J. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.

Tang, X.; Wang, X.; Lv, Z.; Min, Y.; Zhao, W. X.; Hu, B.; Liu, Z.; and Zhang, Z. 2025. Unlocking General Long Chain-of-Thought Reasoning Capabilities of Large Language Models via Representation Engineering. *arXiv preprint arXiv:2503.11314*.

Team, G.; Mesnard, T.; Hardin, C.; Dadashi, R.; Bhupatiraju, S.; Pathak, S.; Sifre, L.; Rivière, M.; Kale, M. S.; Love, J.; et al. 2024. Gemma: Open models based on gemini research and technology. *URL https://arxiv. org/abs/2403.08295*, 2: 10–19.

von Rütte, D.; Anagnostidis, S.; Bachmann, G.; and Hofmann, T. 2024. A Language Model's Guide Through Latent Space. *arXiv preprint arXiv:2402.14433*.

Wang, K.; Hua, H.; and Wan, X. 2019. Controllable unsupervised text attribute transfer via editing entangled latent representation. *Advances in Neural Information Processing Systems*, 32.

Wang, X.; and Zhou, D. 2024. Chain-of-thought reasoning without prompting. *arXiv preprint arXiv:2402.10200*.

Wang, Z.; Whyte, B.; and Xu, C. 2024. Locating and Extracting Relational Concepts in Large Language Models. In *Findings of the Association for Computational Linguistics ACL 2024*, 4818–4832.

Wang, Z.; and Xu, C. 2025. ThoughtProbe: Classifier-Guided Thought Space Exploration Leveraging LLM Intrinsic Reasoning. *arXiv preprint arXiv:2504.06650*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837.

Welling, M.; and Teh, Y. W. 2011. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, 681–688. Citeseer.

Xiao, Y.; Chaoqun, W.; Zhang, Y.; Wang, W.; Lin, B.; He, X.; Shen, X.; and Ye, J. 2024. Enhancing Multiple Dimensions of Trustworthiness in LLMs via Sparse Activation Control. *Advances in Neural Information Processing Systems*, 37: 15730–15764.

Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T.; Cao, Y.; and Narasimhan, K. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36: 11809–11822.

Zelikman, E.; Wu, Y.; Mu, J.; and Goodman, N. 2022. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35: 15476–15488.

Zhang, X.; Du, C.; Pang, T.; Liu, Q.; Gao, W.; and Lin, M. 2024. Chain of Preference Optimization: Improving Chain-of-Thought Reasoning in LLMs. *arXiv preprint arXiv:2406.09136*.

Zhu, M.; Weng, Y.; Yang, L.; and Zhang, Y. 2024. Personality alignment of large language models. *arXiv preprint arXiv:2408.11779*.

Zou, A.; Phan, L.; Chen, S.; Campbell, J.; Guo, P.; Ren, R.; Pan, A.; Yin, X.; Mazeika, M.; Dombrowski, A.-K.; et al. 2023. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*.