ERRV: Eliciting Efficient Reasoning through Reasoning Vectors for Policy Optimization in Large Language Models

Zhuowen Han, Lei Yang, Renren Jin, Dan Shi, Chenxi Sun, Deyi Xiong*
TJUNLP Lab, School of Computer Science and Technology, Tianjin University, China
{zwhan, dyxiong}@tju.edu.cn

Abstract

Recently, large reasoning models have achieved impressive performance, but their lengthy reasoning processes incur substantial inference overhead. To mitigate this issue, we propose the concept of reasoning vectors, representations extracted from the model's hidden states, which can guide the model towards generating more concise and accurate responses. Building upon this, we present ERRV, a training framework that elicits **e**fficient **r**easoning through **r**easoning **v**ectors, which enables the model to generate high-quality responses during reinforcement learning. By performing targeted policy optimization on both accuracy and length objectives, ERRV effectively activates the model's latent capability for efficient reasoning. Our experiments demonstrate that after training with ERRV, the model achieves approximately 30% reduction in reasoning length while maintaining stable accuracy, without guidance from the reasoning vector during inference. This establishes a trade-off between efficiency and performance. Furthermore, we identify key properties of reasoning vectors: robustness, characterized by high similarity before and after training, and generalizability, demonstrating applicability across base models, distilled models, RL-trained models, parameter-merged models, and mixed-thought models. These properties collectively guarantee the reliability and broad applicability of our approach.

1 Introduction

Recent advances in large language models (LRMs) have enabled reasoning models such as OpenAI o3 (OpenAI, 2025) and DeepSeek-R1 (DeepSeek-AI, 2025), which employ extended chain-of-thought processes to achieve remarkable performance on complex reasoning tasks (Zhang and Xiong, 2025; Jin et al., 2025; Liu et al., 2025b) through iterative exploration, self-reflection (Venhoff et al., 2025), and error correction. However, these lengthy reasoning processes substantially increase inference overhead and latency. Prior research suggests that reasoning length inflation may be uncorrelated with correctness, and that concise reasoning chains can actually yield superior accuracy (Hassid et al., 2025). Addressing this trade-off between reasoning capability and efficiency is critical for real-world deployment.

Existing efforts to improve the reasoning efficiency of models mainly focus on modifying the reinforcement learning reward, such as adding length penalties (Aggarwal and Welleck, 2025; Luo et al., 2025a), reconstructing datasets into more concise formats or into positive–negative pairs for training (Chen et al., 2024; Xia et al., 2025; Munkhbat et al., 2025; Yang et al., 2025), and employing prompt budgets to control generation length (Han et al., 2025a; Xu et al., 2025). Most of these methods operate externally rather than targeting the internal states of LRMs. However, we argue that the efficient reasoning capability is intrinsically embedded within models. Recent work on interpreting LRMs' reasoning mechanisms has shown that hidden states of prompts can predict various reasoning properties, including final answers (Ashok and May, 2025), reasoning token counts (Sheng et al., 2025b), response length, reasoning steps, and answer confidence (Dong et al., 2025b). These findings suggest models possess the ability to control response length and accuracy, but this ability is not automatically activated. Therefore, we aim to leverage the hidden states, which contain rich internal information, to trigger this latent capability.

Inspired by studies using steering vectors to modulate reasoning patterns (Venhoff et al., 2025) and improve reasoning accuracy (Højer et al., 2025; Liu et al., 2025a), we propose the concept of **Reasoning Vector** along with a stable method for computing it. Specifically, the reasoning vector is defined as the difference between the hidden states of

*Corresponding author

Findings of the Association for Computational Linguistics: ACL 2026, pages 28557–28570

the shortest correct response and the longest incorrect response. Our experiments demonstrate that this vector can stably improve response accuracy and reduce response length, while remaining stable throughout the reinforcement learning training process.

Reasoning vectors can be utilized in two ways: **(1) Direct inference-time injection:** Similar to traditional steering vectors, reasoning vectors can be directly injected into the model's hidden states during inference to guide generation. However, this approach offers limited efficiency improvements and undermines the model's autonomy and consistency. **(2) Training-time integration:** To address these limitations and fully elicit the model's efficient reasoning capability, we propose ERRV, a training framework that integrates reasoning vectors into reinforcement learning. By guiding the rollout process with the vector, we obtain shorter and more accurate samples than without the vector, which provide more targeted policy optimization and stronger supervisory signals. Furthermore, to address the train-inference inconsistency caused by introducing the vector during training, we employ importance sampling to ensure unbiased optimization, thereby guaranteeing training stability. Experimental results show that by computing and updating the reasoning vector with only a minimal amount of data at selected training steps, ERRV achieves approximately 30% reduction in reasoning length while maintaining stable accuracy, even without explicit reasoning vector guidance during inference. This establishes a superior trade-off between efficiency and reasoning accuracy.

In addition, we directly apply the reasoning vector to models trained through various approaches and find that it exhibits strong generalization capabilities, proving effective not only for base models and distilled models, but also for models trained through diverse methods, parameter-merged models, and adaptive reasoning models.

In summary, our contributions are as follows:

(1) We propose the concept of reasoning vector along with an efficient computation method based on difficulty sampling.

(2) We introduce ERRV, a novel training framework that performs policy optimization through reasoning vectors, significantly reducing reasoning length while maintaining accuracy.

(3) We analyze three key properties of reasoning vectors: improving accuracy while reducing length, robustness, and generalization.

2 Related Work

Efficient Reasoning Large reasoning models (LRMs) generate lengthy chains of thought, leading to substantial computational costs and latency, prompting exploration of optimization strategies. Most existing methods for improving the efficiency of LRMs focus on reducing response tokens. Some approaches incorporate length-based rewards into reinforcement learning to encourage more concise outputs (Aggarwal and Welleck, 2025; Luo et al., 2025a). Other approaches construct more concise datasets to train the model (Chen et al., 2024; Xia et al., 2025; Munkhbat et al., 2025; Yang et al., 2025). In addition, some works aim to reduce response length without further training, such as by specifying a token limit in the prompt (Han et al., 2025a; Xu et al., 2025). However, these studies overlook that the internal states of LRMs contain valuable information for guiding efficient reasoning. For instance, hidden states can predict final answers (Ashok and May, 2025), and the number of reasoning tokens can be predicted from question activations alone using linear probes, suggesting that LRMs estimate required reasoning strength in advance (Sheng et al., 2025b).

Steering Vector Steering Vector (also known as activation engineering) (Turner et al., 2023; Zou et al., 2023) is an important technique for controlling model behavior by leveraging its internal hidden states (Han et al., 2025b; Dong et al., 2025a). Some preliminary studies have applied this technique to reasoning models. For example, using steering vectors to steer reasoning patterns (Venhoff et al., 2025) and improve reasoning performance (Højer et al., 2025; Liu et al., 2025a). However, these approaches remain underexplored, offer limited improvements in inference efficiency, face challenges in determining optimal scaling factors, and require injecting steering vectors during inference, which undermines the model's autonomy and consistency. Therefore, we propose reasoning vectors with an efficient computation method and integrate them into the reinforcement learning process to elicit the model's reasoning capabilities.

3 Methodology

First, we propose the concept of reasoning vectors, representations extracted from the model's hidden states that encode reasoning-enhancing information. These vectors can be utilized in two

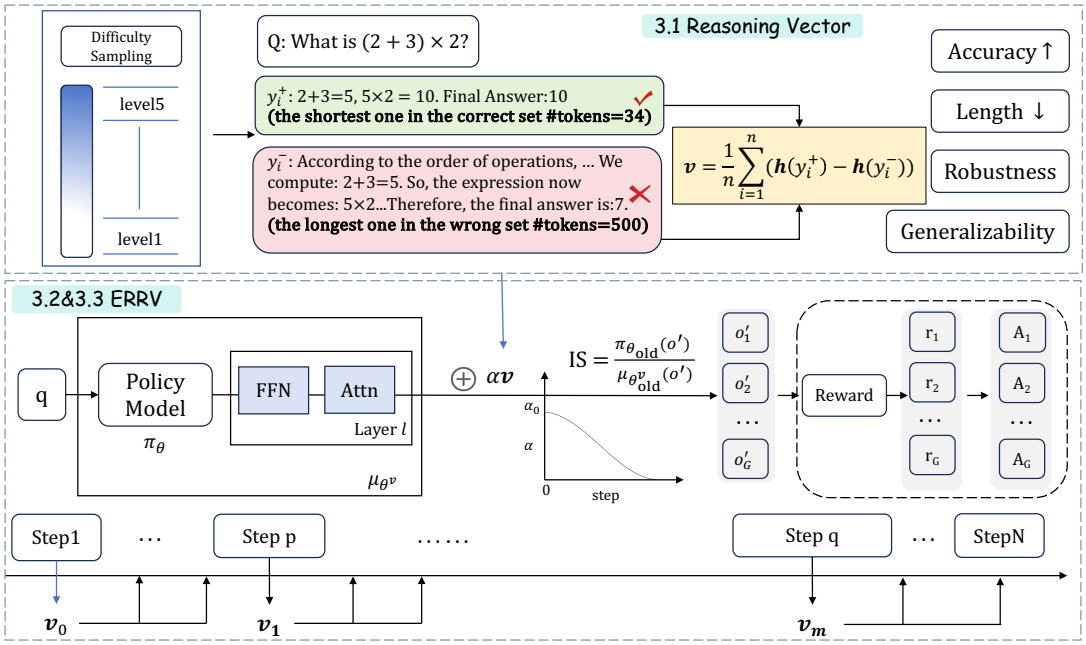


Figure 1: Diagram of the proposed ERRV. Upper: reasoning vector computation from contrastive response pairs. Lower: ERRV training process, where reasoning vectors are periodically computed and updated, then injected during rollout to improve sampling quality, while importance sampling ensures unbiased optimization.

ways: (§3.1) directly injecting into the model during inference to guide the model toward generating accurate and concise outputs, and (§3.2) integrating into training through our proposed ERRV framework, which improves rollout quality and enables the model to learn the reasoning-enhancing information encoded in the vectors. Finally, we incorporate importance sampling weights into the optimization objective to ensure unbiased estimation and training stability(§3.3). The framework is illustrated in Figure 1.

3.1 Reasoning Vectors

The steering vector is a linear direction extracted from the hidden states of LLMs, which can be used to control the model's behavior (Subramani et al., 2022). A simple yet effective technique for computing steering vectors is Mean Difference (MD), which involves constructing contrastive pairs that differ in a specific concept and computing the difference in their mean activations (Tigges et al., 2023; Turner et al., 2023; Arditi et al., 2024). For example, in the domain of sentiment transfer, sentences expressing the emotion "happiness" can serve as positive samples, while sentences expressing the emotion "sadness" can serve as negative samples. By computing the difference between their hidden states, we can extract a vector that controls the expression of the "happiness".

We observe that in reasoning models, among multiple sampled answers to a given question, there are both high-quality and low-quality responses. The ideal answer is one that is both correct and concise, while the least desirable is one that is incorrect and verbose. Motivated by this observation, we propose the concept of **reasoning vectors**—steering vectors specifically designed for reasoning tasks that encode the distinction between efficient and inefficient reasoning patterns. For each question x_i, we sample n responses, defining the shortest correct response as the positive sample y_i^+ and the longest incorrect response as the negative sample y_i^-. If the dataset contains n questions, the resulting reasoning vector is given by:

$$\boldsymbol{v} = \frac{1}{n}\sum_{i=1}^{n}\left(\boldsymbol{h}\left(y_i^+\right) - \boldsymbol{h}\left(y_i^-\right)\right), \qquad (1)$$

where $\boldsymbol{h}(y_i)$ represents the average hidden states of each token in y_i at a specific layer. We select the model's middle-to-late layers for this purpose, which aligns with findings from previous interpretability research indicating that extracting and applying the steering vector at the middle-to-late layers of the model is more efficient than other layers (Tigges et al., 2023; Fan et al., 2025). Additionally, averaging the vectors obtained from each sample enhances the robustness of the vector.

To improve the generalizability of reasoning vectors, we preprocess the dataset D as follows. We stratify questions into 5 difficulty levels based on pass@16 scores (lower scores indicate higher difficulty), with each level corresponding to an equal-width interval of 0.2 in $[0, 1]$ and sample questions from each level. The final dataset comprises n valid samples (see Appendix B.3 for more details) .

During model inference, we directly add the normalized vector to the hidden state at the specific layer as follows:

$$\boldsymbol{h} = (\boldsymbol{h} + \alpha \boldsymbol{v}) \frac{\|\boldsymbol{h}\|_2}{\|\boldsymbol{h} + \alpha \boldsymbol{v}\|_2}, \tag{2}$$

where $\boldsymbol{h}$ is the original hidden state at the specific layer, and α is a scaling factor that controls the strength of the vector's influence.

While reasoning vectors can be directly applied during inference as shown above, this requires continuous vector injection and limits model autonomy. § 3.2 presents ERRV, which integrates vectors into reinforcement learning to help the model internalize efficient reasoning patterns, eliminating the need for vector guidance at inference.

3.2 Training with Reasoning Vectors

Let the prompt be q and the response generated by the LLM π_θ (parameterized by θ) be o. The reward for response o is denoted as $R(o)$. The RLVR objective is to maximize the expected reward of responses generated by π_θ, formulated as:

$$J(\theta) = \mathbb{E}_{o \sim \pi_\theta(\cdot|q)} R(o) . \tag{3}$$

To optimize π_θ for maximizing the expected reward, Group Relative Policy Optimization (GRPO; Shao et al., 2024) obviates the need for an additional value function approximator as required in PPO (Schulman et al., 2017). Instead, it uses the average reward of multiple sampled outputs generated in response to the same question as the baseline. Formally, let π_θ sample G responses $\{o_1, o_2, \cdots, o_G\}$ for each prompt q, and let Q denote the prompt dataset. The optimization objective of GRPO with token- level loss and without the KL penalty term is:

$$\begin{aligned}\mathcal{J}(\theta) =& \mathbb{E}_{q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(O|q)} \\ & \left[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \min \Big(r_{i,t}(\theta) \hat{A}_{i,t}, \right. \\ & \left. \text{clip}(r_{i,t}(\theta), 1-\varepsilon, 1+\varepsilon) \hat{A}_{i,t} \Big) \right],\end{aligned} \tag{4}$$

where $r_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t}|q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q, o_{i,<t})}$, the advantage is given by $\hat{A}_{i,t} = \frac{R(\boldsymbol{o}^i) - \text{mean}\left(\{R(\boldsymbol{o}^j)\}_{j=1}^{G}\right)}{\text{std}\left(\{R(\boldsymbol{o}^j)\}_{j=1}^{G}\right)}$, and ε is a hyperparameter controlling the clipping range.

We propose ERRV, a simple yet effective method for targeted policy optimization that steers the model towards desirable response properties through reasoning vector guidance. Specifically, we first extract a reasoning vector $\boldsymbol{v}$ from a small dataset D following the approach described in Section 3.1. This vector is then injected into the policy during the rollout phase to guide the generation of high-quality responses.

Formally, let $\mathcal{S}(\cdot; \boldsymbol{v}, \alpha, l, n)$ denote the steering operation that modifies the hidden states at layer l according to Equation 2. During rollout, instead of sampling responses $\{o_1, o_2, \cdots, o_G\}$ directly from π_θ, we apply the steering operation to obtain steered responses $\{o'_1, o'_2, \cdots, o'_G\}$:

$$o'_i \sim \mu_{\theta^v}(\cdot|q; \mathcal{S}(\cdot; \boldsymbol{v}, \alpha, l, n)), \quad i = 1, \cdots, G, \tag{5}$$

where μ_{θ^v} denotes the rollout policy guided by the reasoning vector for sampling responses. These steered responses are then used to compute the advantage and update the policy. The ERRV objective can be obtained by simply replacing o with o' in the GRPO objective (Equation 4):

$$\begin{aligned}\mathcal{J}_{\text{ERRV}}(\theta) =& \mathbb{E}_{q \sim Q, \{o'_i\}_{i=1}^{G} \sim \mu_{\theta^v_{\text{old}}}(O'|q; \mathcal{S})} \\ & \left[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|o'_i|} \sum_{t=1}^{|o'_i|} \min \Big(r'_{i,t}(\theta) \hat{A}'_{i,t}, \right. \\ & \left. \text{clip}(r'_{i,t}(\theta), 1-\varepsilon, 1+\varepsilon) \hat{A}'_{i,t} \Big) \right],\end{aligned} \tag{6}$$

where $r'_{i,t}(\theta) = \frac{\pi_\theta(o'_{i,t}|q, o'_{i,<t})}{\pi_{\theta_{\text{old}}}(o'_{i,t}|q, o'_{i,<t})}$, and the advantage $\hat{A}'_{i,t}$ is computed based on the rewards of the steered responses $\{o'_1, \cdots, o'_G\}$.

During training, the model parameters θ are continuously updated at each step. Ideally, $\boldsymbol{v}$ should be recomputed after every step to reflect the latest model state. However, since adjacent training steps typically induce only minor parameter changes, frequent recomputation is computationally expensive and yields diminishing returns.

To balance efficiency and performance, we adopt a periodic update strategy: we recompute the reasoning vector every $k\%$ of the total training steps. This approach maintains alignment between the reasoning vector and the evolving policy while keeping computational overhead manageable.

To enable the model to internalize the reasoning vector during training, we adopt a cosine annealing strategy that gradually reduces the dependence on the reasoning vector by decreasing the scaling factor from α_0 to 0 following a cosine curve:

$$\alpha = 0.5\alpha_0 \times \left(1 + \cos\left(\pi \cdot \frac{s}{ts}\right)\right), \quad (7)$$

where s is the index of the current training step and ts is the total number of training steps.

3.3 Importance Sampling

The reasoning vector transforms the original responses o into higher-quality responses o' that are more accurate and concise. A natural concern is whether these steered responses, being sampled off-policy, introduce significant distribution shift that could destabilize training. Although this shift is minimal, considering that reinforcement learning is sensitive to train-test mismatch (Yao et al., 2025; Zheng et al., 2025), we incorporate importance sampling weights $\frac{\pi_{\theta_{\text{old}}}(o')}{\mu_{\theta^v_{\text{old}}}(o')}$ into the policy gradient estimation:

$$\begin{aligned} J(\theta) &= \mathbb{E}_{o\sim\pi_\theta(\cdot|q)} R(o) \\ &= \mathbb{E}_{o\sim\pi_{\theta_{\text{old}}}(\cdot|q)} \frac{\pi_\theta(o)}{\pi_{\theta_{\text{old}}}(o)} R(o) \\ &= \mathbb{E}_{o'\sim\mu_{\theta^v_{\text{old}}}(\cdot|q;S)} \left[\frac{\pi_{\theta_{\text{old}}}(o')}{\mu_{\theta^v_{\text{old}}}(o')} \frac{\pi_\theta(o')}{\pi_{\theta_{\text{old}}}(o')} R(o') \right], \end{aligned} \quad (8)$$

which corrects for the distribution mismatch between steered and unsteered policies. This ensures that the gradient estimates remain unbiased even when the distribution shift is non-negligible, further enhancing training stability (See Appendix A.1 for details).

4 Experiments

We conducted extensive experiments to examine the effectiveness and properties of reasoning vectors, to validate the proposed ERRV and to discover the underlying mechanisms.

4.1 Setup

Models We selected DeepSeek-R1-Distill-Qwen-1.5B (DS-1.5B) and DeepSeek-R1-Distill-Qwen-7B (DS-7B), demonstrating strong performance on math problem solving, as the initial policy models.

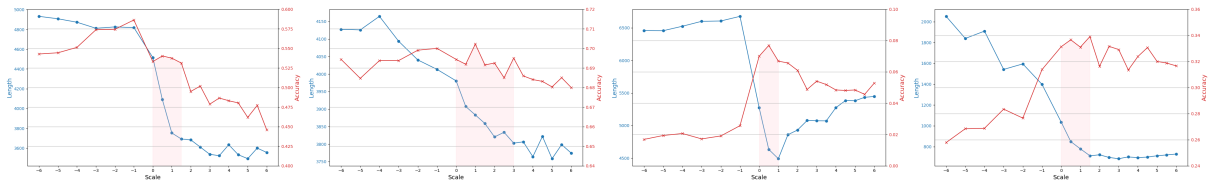
Dataset and Metrics We used the DeepScaleR dataset (Luo et al., 2025b) as our training data , which consists of approximately 40,000 unique problem-answer pairs compiled from AIME 1984-2023, AMC, Omni-MATH (Gao et al., 2025), and STILL (Min et al., 2024). Additionally, we randomly sampled 1,000 samples from the DeepScaleR dataset to construct the dataset D for computing the reasoning vector. For evaluation, we used three datasets of increasing difficulty: GSM8K (Cobbe et al., 2021), MATH500 (Lightman et al., 2024), and AIME 2024 (30 Olympiad-level mathematics problems) to assess the performance of methods. We assessed performance using two metrics: accuracy and response length. Given the limited size of AIME 2024, we sampled 16 responses for each problem and reported the average results. For all models, we conducted experiments with evaluation context sizes of 8K tokens, and set the temperature to 0.6 as recommended in DeepSeek's model cards.

Implementation Details We implemented ERRV based on the VeRL framework (Sheng et al., 2025a). We used a context size of 8K tokens, batch size of 128, and learning rate of 1×10^{-6}. The reasoning vector was recomputed every $k = 6\%$ of training steps using $n = 10$ samples. The scaling factor α_0 was set to 1.0 for DS-1.5B and 3.0 for DS-7B (Section 4.3). The comparison of using different n is shown in Section 4.5. All models were trained for 1 epoch (314 steps in total).

4.2 Baselines

Existing methods can be categorized into four main approaches: parameter fusion, length penalty, data reconstruction, and adaptive thinking strategies. We selected one representative method from each category as our baseline.

- **ModelMerging** (Wu et al., 2025b) reduces response length by parameter fusion between reasoning models and non-reasoning models.
- **TLMRE** (Arora and Zanette, 2025) penalizes the length of correct responses to train the model to produce correct solutions with the minimum number of tokens.
- **DPO** constructs preference data by sampling multiple responses for each problem and pairing the shortest correct response with the longest response, then uses DPO (Rafailov et al., 2023) to finetune the model.



(a) DS-1.5B (b) DS-7B (c) Qwen-2.5-1.5B (d) Qwen-2.5-7B

Figure 2: Effect of v under different scaling factors across different models. Blue: average response length; Red: accuracy on test set. Layer 20 is used as the injection layer (all models have 28 layers).

- **AdaptThink** (Zhang et al., 2025a) teaches models to adaptively choose the optimal thinking mode based on problem difficulty.

For fair comparison, all baselines were trained on DeepScaleR dataset. Note that AdaptThink achieves length reduction by selectively skipping reasoning for certain problems, while other methods retain thinking for all cases, making direct comparison inappropriate. We report its results for reference but exclude it from main comparisons.

4.3 The performance of reasoning vectors

First, we investigated the effectiveness and properties of reasoning vectors. To demonstrate their effectiveness across different models, we additionally evaluated Qwen-2.5-1.5B and Qwen-2.5-7B. We computed reasoning vectors for each model using $n = 200$ samples and applied them during inference with varying scaling factors $\alpha \in [-6, 6]$ according to Equation 2. We randomly selected 400 examples from DeepScaleR as the test set. The results are shown in Figure 2. We observed that for all models, within a certain range of α:

> Reasoning vectors can improve the accuracy and reduce the length of responses.

The effective range of α typically lies within $[0, \alpha_x]$ (corresponding to the pink shaded area in Figure 2), where applying the reasoning vector improves or maintains model accuracy. This range represents the ideal scaling factor for v, with α_x varying across models depending on their sensitivity to v. For instance, $\alpha_x \approx 1.0$ for DS-1.5B, while $\alpha_x \approx 3.0$ for DS-7B. In ERRV training, we use $\alpha_0 = \alpha_x$ as the initial value of the scaling factor.

Next, we investigated the stability of reasoning vectors before and after RL training. We selected two model pairs: Qwen2.5-1.5B with its GRPO-trained version Qwen2.5-1.5B-Simple-Zoo, and Qwen2.5-7B with its corresponding version Qwen2.5-7B-Simple-Zoo (Zeng et al., 2025).

First, we extracted reasoning vectors from each model and computed the cosine similarity between vectors from pre-training and post-training models, yielding 0.834 for the 1.5B pair and 0.828 for the 7B pair. These high similarity scores indicate that:

> Reasoning vectors remain stable across RL training.

We then conducted cross-application experiments (see Figure 6 in Appendix): for each model, we applied both its own reasoning vector (blue curves) and the vector from its counterpart (red curves). The results reveal a consistent pattern: for base models (Figure 6(a), 6(b)), vectors extracted from the RL-trained counterparts (red curves) outperform self-extracted vectors (blue curves), maintaining higher accuracy and shorter lengths across a wider range of α values. For RL-trained models (Figure 6(c), 6(d)), vectors from the RL versions (blue curves) similarly demonstrate superior performance. This indicates that:

> Reinforcement learning makes the reasoning vectors more robust.

These findings provide a solid theoretical and empirical foundation for developing ERRV.

4.4 The performance of ERRV

Table 1 presents the evaluation results of different methods, as well as the results of applying reasoning vectors during inference only (§3.1) on top of these methods. The last two columns show the average accuracy change and average length reduction percentage, respectively.

Since different scenarios require different trade-offs between length and accuracy, we established threshold-based criteria rather than a single com-

Table 1: Accuracy (ACC) and response length (LEN) on three math benchmarks. Gray rows show results with reasoning vectors applied at inference to the baseline above. Light gray (AdaptThink) is excluded from comparison. Bold indicates the best results; red indicates the global best (including vector-augmented results).

Method	GSM8K		MATH 500		AIME 2024		AVG	
	ACC ↑	LEN ↑	ACC ↑	LEN ↑	ACC ↑	LEN ↓	ACC ↑	LEN ↓
DeepSeek-R1-Distill-Qwen-1.5B								
Original	77.9	955	75.4	3662	20.8	7333	-	-
+vector	76.4	698	79.4	3554	23.5	7347	+1.7	-9.9%
ModelMerging	77.9	426	72.6	1291	14.3	3679	-3.1	-55.3%
+vector	79.8	440	76.0	1133	14.4	3301	-1.3	-59.3
TLMRE	71.3	306	72.4	1394	21.5	5170	-3.0	-53.1%
+vector	70.1	299	74.4	1360	26.3	4962	-1.1	-54.6%
DPO	76.4	1167	80.6	3755	26.5	7179	+3.1	+7.5%
+vector	74.5	739	80.6	3566	26.7	7066	+2.6	-9.6%
AdaptThink	80.5	424	76.2	1447	18.3	4077	+0.3	-53.5%
+vector	79.2	350	77.2	1018	20.2	3454	+0.8	-62.8%
ERRV	79.0	720	79.6	2175	22.1	4888	**+2.2**	**-32.9%**
+vector	77.9	573	84.4	2081	22.1	4793	+3.4	-39.3%
ERRV-IS	82.1	824	83.0	2598	28.5	6426	+6.5	-18.4%
+vector	81.2	705	81.6	2406	27.3	6198	+5.3	-25.3%
DeepSeek-R1-Distill-Qwen-7B								
Original	86.7	659	88.4	3171	39.8	6862	-	-
ModelMerging	77.3	465	78.4	1310	24.8	4162	-11.5	-42.5%
TLMRE	88.6	749	90.2	2306	46.9	5922	+3.6	-9.1%
DPO	86.3	844	84.2	3031	33.5	6684	-3.6	+7.0%
AdaptThink	91.1	499	87.6	1184	45.6	5280	+3.1	-36.7%
ERRV	87.3	558	86.0	2069	33.8	5558	**-2.6**	**-23.0%**
ERRV-IS	88.5	730	91.0	2301	50.2	5783	+4.9	-10.8%

bined metric. Methods must achieve at least 20% length reduction while limiting accuracy loss to no more than 2% for DS-1.5B or 3% for DS-7B. We set a more relaxed accuracy threshold for the larger model due to its higher baseline performance.

Looking at the DS-1.5B results in the white rows, only ERRV meets our criteria among all baselines (AdaptThink is excluded from comparison due to methodological differences as discussed in Section 4.2). ERRV achieves a 32.9% length reduction while simultaneously improving accuracy by 2.2%. In contrast, other methods either achieve substantial length reduction at the cost of excessive accuracy loss, or fail to reduce length sufficiently to meet our optimization objectives.

The gray rows show results when applying reasoning vectors during inference on top of these methods. All methods exhibit significant improvements over their base versions, demonstrating that reasoning vectors are effective not only for base models and distilled models, but also for models trained through various methods, parameter-merged models, and adaptive reasoning models. Noteworthy is AdaptThink with reasoning vector guidance, which achieves exceptional results: 62.8% length reduction with 0.8% accuracy gain, representing an additional 9.3% length reduction and 0.5% improvement over vanilla AdaptThink.

Comparing the second row (direct reasoning vector application) with ERRV reveals a substantial performance gap, indicating that ERRV successfully elicits the efficient reasoning capability, enabling the model to maintain excellent performance even without vector guidance during inference.

We also observed that ERRV with importance sampling (ERRV-IS) achieves more stable training and the highest accuracy improvement, though with reduced length compression below our threshold. This variant is better suited for scenarios that prioritize accuracy over aggressive length reduction.

The lower section shows DS-7B results, which follow similar patterns. Only ERRV meets our criteria, achieving 23% length reduction with 2.6% accuracy decrease. Other insights remain consistent with DS-1.5B and are not elaborated here.

Out-of-Distribution Performance Apart from mathematical reasoning, we further conducted experiments on code generation and general knowledge reasoning. We evaluated the models be-

Table 2: Accuracy (ACC) and response length (LEN) on LiveCodeBench-v6 and MMLU (DS-1.5B).

Model	LiveCodeBench-v6		MMLU	
	ACC	LEN	ACC	LEN
Original	14.5	6026	42.9	1357
ERRV	22.3	4442	43.3	669

fore and after ERRV training on LiveCodeBench-v6 (Jain et al., 2025) and MMLU (Hendrycks et al., 2021), with results shown in the Table 2.

On LiveCodeBench-v6, ERRV reduces the average generated length from 6,026 to 4,442 tokens while improving accuracy by 7.8%. On MMLU, ERRV nearly halves the reasoning length from 1,357 to 669 tokens while maintaining stable accuracy with a marginal improvement. These results demonstrate that ERRV is not limited to mathematical reasoning and exhibits strong generalization across diverse task types.

4.5 Ablation Study

First, we investigated the impact of the scaling factor α of the reasoning vector on training outcomes. We examined three adjustment strategies: (1) **Cosine Decay:** α decays following a cosine schedule according to Equation 7. (2) **Constant:** α remains fixed at the initial value α_0 throughout the training process. (3) **Dynamic:** Starting from α_0, α is dynamically adjusted based on reward on the test set. Specifically, α is increased by δ if the reward rises and decreased by δ otherwise, with δ set to 0.1.

We evaluated these strategies for both ERRV and ERRV-IS, with the results illustrated in Figure 3. We observed that the Cosine Decay and Dynamic strategies yield similar performance. In contrast, the Constant strategy results in a substantial reduction in response length accompanied by a significant drop in accuracy. This indicates that α should gradually decrease during training to enable the model to effectively internalize the information from the reasoning vector. Additionally, with importance sampling, the variation pattern of the scaling factor has minimal impact on the results.

Furthermore, we analyzed the impact of the sample size n used for vector computation on the results. As shown in Figure 4, increasing n does not lead to significant changes in length reduction or accuracy improvement. This demonstrates that using a small batch of samples is sufficient

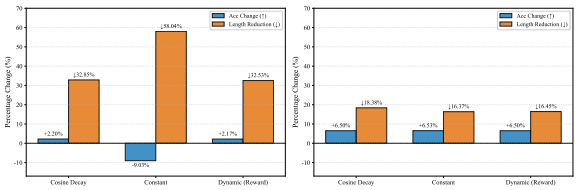


Figure 3: Performance of ERRV and ERRV-IS (DS-1.5B) under different α adjustment strategies

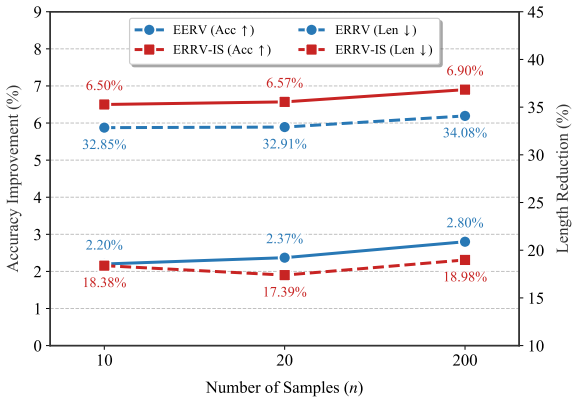


Figure 4: Performance of ERRV and ERRV-IS (DS-1.5B) using different number of samples to compute reasoning vector.

to compute effective vectors, thereby significantly reducing the additional computational overhead of training (See Appendix B.2 for detailed results and Appendix B.3 for cost analysis).

4.6 Analysis of Rollout Quality

We conducted a detailed analysis of the quality of responses generated during the rollout of training.

First, Figure 5 compares the average response length generated during rollouts by ERRV versus standard GRPO. It is evident that ERRV generates significantly shorter responses than GRPO, with the gap gradually widening as training progresses.

Next, to isolate the impact of reasoning vectors, we added an unsteered rollout phase during ERRV training. At each training step, alongside the standard steered rollout, we performed an additional unsteered rollout (without reasoning vectors) for comparison. Figure 7 in Appendix plots the difference in average length between the unsteered responses and the steered responses (ERRV). The consistently positive values confirm that reasoning vectors effectively reduce the length of generated responses at every step.

These analyses demonstrate that our reasoning vector effectively enhances sampling quality during training, thereby facilitating more targeted policy

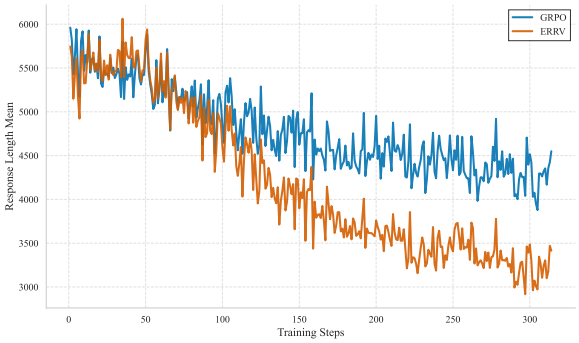


Figure 5: Comparison of average rollout response length on DS-1.5B: ERRV vs. GRPO.

Table 3: Average number of reasoning steps.

Model	GSM8K	MATH	AIME
DS-1.5B	25.04	103.11	231.61
DS-1.5B-ERRV	15.54	46.37	113.62

optimization with stronger supervisory signals.

4.7 Analysis of Reasoning Process

Furthermore, we analyzed how ERRV reduces reasoning length from three perspectives: the number of reasoning steps, reasoning redundancy, and reasoning structure.

First, we approximated reasoning steps using a paragraph-based approach, where segments separated by `\n\n` are each treated as a single step (Wu et al., 2025a; Zhang et al., 2025b). We extracted the thinking content from model responses, segmented them accordingly, and computed the average number of steps per response (Table 3). The results show that ERRV-trained models produce responses with significantly fewer reasoning steps.

Second, a substantial portion of long thinking process arises from rollback, self-doubt, recalculation, and repeated verification. These behaviors are often signaled by discourse markers such as "Wait", "Let me check", "Actually" and "I made a mistake." We therefore counted the occurrences of these tokens in the model's reasoning traces, and report the average frequency per response in Table 4. The results show that ERRV substantially reduces the frequency of these trial-and-error behaviors.

Third, responses to math problems typically consist of five reasoning components (Wu et al., 2025a): (1) Pivotal Reasoning, (2) Productive Elaboration and Calculation, (3) Exploring Alternatives, (4) Verification and Self-Correction, and (5) Non-Substantive Statements, where (1)-(2) are core steps and (3)-(5) are auxiliary and often redundant. Detailed descriptions are provided in Appendix B.5. We annotated each sentence in the model's reasoning traces on MATH500 using GPT-4.1 and computed the proportion of each reasoning type per response, as shown in Table 5.

Table 4: Average number of discourse markers.

Model	GSM8K	MATH	AIME
DS-1.5B	3.39	15.98	43.04
DS-1.5B-ERRV	0.49	2.53	7.59

Table 5: Average proportion of each reasoning type.

Model	(1)	(2)	(3, 4, 5)
DS-1.5B	28.76%	30.94%	40.30%
DS-1.5B-ERRV	30.62%	33.61%	35.77%

The results reveal clear shifts in reasoning composition. Compared to DS-1.5B, DS-1.5B-ERRV allocates a higher proportion to core steps (1) and (2), reflecting a stronger focus on essential reasoning and computation, while auxiliary steps (3)–(5) decline, indicating fewer digressions and less redundant self-checking. Additionally, we measured the reduction in token counts for each reasoning component. We've found that token counts decrease by 9.38% in (1), 18.16% in (2), and 25.79% in (3)-(5), confirming that reductions are concentrated in the less essential components.

Overall, ERRV effectively optimizes the reasoning structure by reducing redundant steps while reinforcing the importance of key steps.

5 Conclusion

We have presented Reasoning Vector, which reduces response length while improving accuracy. Building upon this, we have proposed ERRV, a novel reinforcement learning algorithm that leverages reasoning vectors to guide the model toward generating high-quality responses during rollout, thereby enabling more targeted policy optimization. Experimental results demonstrate that ERRV significantly reduces inference costs by optimizing the reasoning structure, while maintaining competitive accuracy and exhibiting strong generalization across tasks. Furthermore, we investigate various properties of reasoning vectors, which not only validates the stability of ERRV but also provides new insights for future research.

Limitations

Due to limited computational resources, we only conducted experiments on 1.5B and 7B models. Nevertheless, these experiments demonstrate the effectiveness of ERRV across different model sizes. Additionally, our experiments primarily focus on mathematical reasoning tasks. While we have validated the generalizability of ERRV on code generation and general knowledge reasoning, future work could further explore its application to other domains and task types.

Acknowledgements

The present research was supported by the National Key Research and Development Program of China (Grant No. 2024YFE0203000). We would like to thank the anonymous reviewers for their insightful comments.

References

Pranjal Aggarwal and Sean Welleck. 2025. L1: Controlling How Long A Reasoning Model Thinks With Reinforcement Learning. *CoRR*, abs/2503.04697.

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. 2024. Refusal in Language Models Is Mediated by a Single Direction. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Daman Arora and Andrea Zanette. 2025. Training Language Models to Reason Efficiently. *CoRR*, abs/2502.04463.

Dhananjay Ashok and Jonathan May. 2025. Language Models Can Predict Their Own Behavior. *CoRR*, abs/2502.13329.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2024. Do NOT Think That Much for 2+3=? On the Overthinking of o1-Like LLMs. *CoRR*, abs/2412.21187.

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. *CoRR*, abs/2110.14168.

DeepSeek-AI. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *CoRR*, abs/2501.12948.

Weilong Dong, Xinwei Wu, Renren Jin, Shaoyang Xu, and Deyi Xiong. 2025a. CONTRANS: Weak-to-Strong Alignment Engineering via Concept Transplantation. In *Proceedings of the 31st International Conference on Computational Linguistics, COLING 2025, Abu Dhabi, UAE, January 19-24, 2025*, pages 4130–4148. Association for Computational Linguistics.

Zhichen Dong, Zhanhui Zhou, Zhixuan Liu, Chao Yang, and Chaochao Lu. 2025b. Emergent Response Planning in LLMs. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*. OpenReview.net.

Siqi Fan, Xin Jiang, Xiang Li, Xuying Meng, Peng Han, Shuo Shang, Aixin Sun, and Yequan Wang. 2025. Not All Layers of LLMs Are Necessary During Inference. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2025, Montreal, Canada, August 16-22, 2025*, pages 5083–5091. ijcai.org.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, Zhengyang Tang, Benyou Wang, Daoguang Zan, Shanghaoran Quan, Ge Zhang, Lei Sha, Yichang Zhang, Xuancheng Ren, Tianyu Liu, and Baobao Chang. 2025. Omni-MATH: A Universal Olympiad Level Mathematic Benchmark for Large Language Models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. 2025a. Token-Budget-Aware LLM Reasoning. In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 24842–24855. Association for Computational Linguistics.

Zhuowen Han, Xinwei Wu, Dan Shi, Renren Jin, and Deyi Xiong. 2025b. Towards a Unified Paradigm of Concept Editing in Large Language Models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 18445–18461. Association for Computational Linguistics.

Michael Hassid, Gabriel Synnaeve, Yossi Adi, and Roy Schwartz. 2025. Don't Overthink it. Preferring Shorter Thinking Chains for Improved LLM Reasoning. *CoRR*, abs/2505.17813.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring Massive Multitask Language Understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Bertram Højer, Oliver Simon Jarvis, and Stefan Heinrich. 2025. Improving Reasoning Performance in

Large Language Models via Representation Engineering. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2025. LiveCodeBench: Holistic and Contamination Free Evaluation of Large Language Models for Code. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Renren Jin, Pengzhi Gao, Yuqi Ren, Zhuowen Han, Tongxuan Zhang, Wuwei Huang, Wei Liu, Jian Luan, and Deyi Xiong. 2025. Revisiting Entropy in Reinforcement Learning for Large Reasoning Models. *CoRR*, abs/2511.05993.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP 2023, Koblenz, Germany, October 23-26, 2023*, pages 611–626. ACM.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's Verify Step by Step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Sheng Liu, Tianlang Chen, Pan Lu, Haotian Ye, Yizheng Chen, Lei Xing, and James Zou. 2025a. Fractional Reasoning via Latent Steering Vectors Improves Inference Time Compute. *CoRR*, abs/2506.15882.

Yan Liu, Minghui Zhang, Bojian Xiong, Yifan Xiao, Yinong Sun, Yating Mei, Longyu Zeng, Jingchao Yang, Yang Wang, and Deyi Xiong. 2025b. HighMATH: Evaluating Math Reasoning of Large Language Models in Breadth and Depth. In *Findings of the Association for Computational Linguistics: EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 10241–10253. Association for Computational Linguistics.

Haotian Luo, Li Shen, Haiying He, Yibo Wang, Shiwei Liu, Wei Li, Naiqiang Tan, Xiaochun Cao, and Dacheng Tao. 2025a. O1-Pruner: Length-Harmonizing Fine-Tuning for O1-Like Reasoning Pruning. *CoRR*, abs/2501.12570.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Li Erran Li, and 1 others. 2025b. Deepscaler: Surpassing O1-preview with a 1.5B Model by Scaling Rl. *Notion Blog*.

Yingqian Min, Zhipeng Chen, Jinhao Jiang, Jie Chen, Jia Deng, Yiwen Hu, Yiru Tang, Jiapeng Wang, Xiaoxue Cheng, Huatong Song, Wayne Xin Zhao, Zheng Liu, Zhongyuan Wang, and Ji-Rong Wen. 2024. Imitate, Explore, and Self-Improve: A Reproduction Report on Slow-thinking Reasoning Systems. *CoRR*, abs/2412.09413.

Tergel Munkhbat, Namgyu Ho, Seo Hyun Kim, Yongjin Yang, Yujin Kim, and Se-Young Yun. 2025. Self-Training Elicits Concise Reasoning in Large Language Models. In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 25127–25152. Association for Computational Linguistics.

OpenAI. 2025. Introducing OpenAI o3 and o4-mini.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *CoRR*, abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *CoRR*, abs/2402.03300.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025a. HybridFlow: A Flexible and Efficient RLHF Framework. In *Proceedings of the Twentieth European Conference on Computer Systems, EuroSys 2025, Rotterdam, The Netherlands, 30 March 2025 - 3 April 2025*, pages 1279–1297. ACM.

Leheng Sheng, An Zhang, Zijian Wu, Weixiang Zhao, Changshuo Shen, Yi Zhang, Xiang Wang, and Tat-Seng Chua. 2025b. On Reasoning Strength Planning in Large Reasoning Models. *CoRR*, abs/2506.08390.

Nishant Subramani, Nivedita Suresh, and Matthew E. Peters. 2022. Extracting Latent Steering Vectors from Pretrained Language Models. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 566–581. Association for Computational Linguistics.

Curt Tigges, Oskar John Hollinsworth, Atticus Geiger, and Neel Nanda. 2023. Linear Representations of Sentiment in Large Language Models. *CoRR*, abs/2310.15154.

Alexander Matt Turner, Lisa Thiergart, David Udell, Gavin Leech, Ulisse Mini, and Monte MacDiarmid. 2023. Activation Addition: Steering Language Models Without Optimization. *CoRR*, abs/2308.10248.

Constantin Venhoff, Iván Arcuschin, Philip Torr, Arthur Conmy, and Neel Nanda. 2025. Understanding Reasoning in Thinking Language Models via Steering Vectors. *CoRR*, abs/2506.18167.

Canhui Wu, Qiong Cao, Chang Li, Zhenfang Wang, Chao Xue, Yuwei Fan, Wei Xi, and Xiaodong He. 2025a. Beyond Token Length: Step Pruner for Efficient and Accurate Reasoning in Large Language Models. *CoRR*, abs/2510.03805.

Han Wu, Yuxuan Yao, Shuqi Liu, Zehua Liu, Xiaojin Fu, Xiongwei Han, Xing Li, Hui-Ling Zhen, Tao Zhong, and Mingxuan Yuan. 2025b. Unlocking Efficient Long-to-Short LLM Reasoning with Model Merging. *CoRR*, abs/2503.20641.

Heming Xia, Yongqi Li, Chak Tou Leong, Wenjie Wang, and Wenjie Li. 2025. TokenSkip: Controllable Chain-of-Thought Compression in LLMs. *CoRR*, abs/2502.12067.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. 2025. Chain of Draft: Thinking Faster by Writing Less. *CoRR*, abs/2502.18600.

Wenkai Yang, Shuming Ma, Yankai Lin, and Furu Wei. 2025. Towards Thinking-Optimal Scaling of Test-Time Compute for LLM Reasoning. *CoRR*, abs/2502.18080.

Feng Yao, Liyuan Liu, Dinghuai Zhang, Chengyu Dong, Jingbo Shang, and Jianfeng Gao. 2025. Your Efficient RL Framework Secretly Brings You Off-Policy RL Training.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. 2025. SimpleRL-Zoo: Investigating and Taming Zero Reinforcement Learning for Open Base Models in the Wild. *CoRR*, abs/2503.18892.

Jiajie Zhang, Nianyi Lin, Lei Hou, Ling Feng, and Juanzi Li. 2025a. AdaptThink: Reasoning Models Can Learn When to Think. *CoRR*, abs/2505.13417.

Shaowei Zhang and Deyi Xiong. 2025. BackMATH: Towards Backward Reasoning for Solving Math Problems Step by Step. In *Proceedings of the 31st International Conference on Computational Linguistics, COLING 2025 - Industry Track, Abu Dhabi, UAE, January 19-24, 2025*, pages 466–482. Association for Computational Linguistics.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025b. The Lessons of Developing Process Reward Models in Mathematical Reasoning. In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, Findings of ACL, pages 10495–10516. Association for Computational Linguistics.

Chujie Zheng, Kai Dang, Bowen Yu, Mingze Li, Huiqiang Jiang, Junrong Lin, Yuqiong Liu, An Yang, Jingren Zhou, and Junyang Lin. 2025. Stabilizing Reinforcement Learning with LLMs: Formulation and Practices. *arXiv preprint arXiv:2512.01374*.

Andy Zou, Long Phan, Sarah Li Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, and 2 others. 2023. Representation Engineering: A Top-Down Approach to AI Transparency. *CoRR*, abs/2310.01405.

A Supplement to Method

A.1 Detail of Importance Sampling

Let us discuss the importance sampling in detail. In the original GRPO, the importance sampling ratio is defined as

$$r_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t})}{\pi_{\theta_{\text{old}}}(o_{i,t})}. \tag{9}$$

However, when we introduce the reasoning vector v into the inference engine, the model's output distribution shifts from $\pi_\theta(o|q)$ to $\mu_{\theta^v}(o'|q;\mathcal{S})$. This causes a train-inference mismatch problem when directly applying the original ratio $r'_{i,t}(\theta) = \frac{\pi_\theta(o'_{i,t})}{\pi_{\theta_{\text{old}}}(o'_{i,t})}$.

To address this issue, we employ importance sampling to correct the distribution shift. Specifically, we introduce an importance sampling weight:

$$w = \frac{\pi_{\theta_{\text{old}}}(o'_{i,t})}{\mu_{\theta^v_{\text{old}}}(o'_{i,t})}. \tag{10}$$

Consequently:

$$\begin{aligned} r^{\text{is}}_{i,t}(\theta) &= w \cdot r'_{i,t}(\theta) \\ &= \frac{\pi_{\theta_{\text{old}}}(o'_{i,t})}{\mu_{\theta^v_{\text{old}}}(o'_{i,t})} \cdot \frac{\pi_\theta(o'_{i,t})}{\pi_{\theta_{\text{old}}}(o'_{i,t})} \\ &= \frac{\pi_\theta(o'_{i,t})}{\mu_{\theta^v_{\text{old}}}(o'_{i,t})}, \end{aligned} \tag{11}$$

This correction ensures training stability while maintaining the unbiasedness of the policy gradient estimator.

B Experiment Details

B.1 Cross-application Experiments

The results of cross-application experiments is shown in Figure 6.

B.2 Detailed Data in Ablation Experiments

Table 6 is the detailed results of ERRV and ERRV-IS under different α adjustment strategies and different number of samples to compute reasoning vector, using DS-1.5B as an example.

B.3 Cost Analysis

We detail the computation of reasoning vectors and their associated time cost.

For each sample, we generated 16 responses using the vLLM framework (Kwon et al., 2023) for fast inference. Questions with extreme performance (100% or 0% pass rate) were excluded as they cannot form valid contrastive pairs. When this occurred, we resampled from the corresponding difficulty level until obtaining valid examples. To ensure precision, we re-ran inference using the native model to extract accurate hidden states for the computation described in Section 3.1.

For our experimental setup with 10 samples for computing reasoning vectors (including resampling for invalid samples), this process took approximately 5 minutes on a single A100 GPU. If we update the reasoning vectors at $k\% = 6\%$ of total training steps, which corresponds to approximately 18 updates, the total overhead is only 90 minutes. It is entirely acceptable.

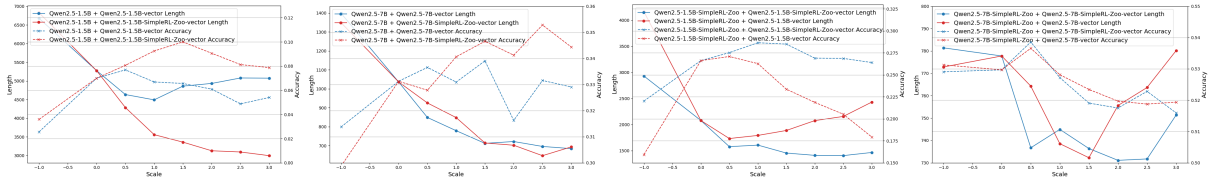
B.4 Add an Unsteered Rollout

The difference in average length between the unsteered responses and the steered responses (ERRV) is shown in Figure 7.

B.5 Detailed Descriptions of Mathematical Reasoning Components

Responses to math problems typically consist of the following components (Wu et al., 2025a), with the first two representing core reasoning steps and the remaining three serving as auxiliary reasoning steps that are often redundant and can be omitted:

- **Pivotal Reasoning** Core steps that directly advance the solution.
- **Productive Elaboration and Calculation** Supporting explanations or detailed arithmetic.
- **Exploring Alternatives** Consideration of different methods or perspectives.
- **Verification and Self-Correction** In-line checks or corrections of prior steps.
- **Non-Substantive Statements** Remarks that do not contribute to the logical solution path.



(a) Qwen2.5-1.5B (b) Qwen2.5-7B (c) Qwen2.5-1.5B-Simple-Zoo (d) Qwen2.5-7B-Simple-Zoo

Figure 6: Cross-application performance of reasoning vectors. Subfigures (a) and (b) show base models (Qwen2.5-1.5B and Qwen2.5-7B) applied with their own vectors (blue) vs. vectors from their GRPO-trained counterparts (red). Subfigures (c) and (d) show GRPO-trained models (Simple-Zoo versions) applied with their own vectors (blue) vs. vectors from base models (red). Solid lines indicate response length; dashed lines indicate accuracy.

Table 6: Detailed results of ERRV and ERRV-IS (DS-1.5B) under different α adjustment strategies and different number of samples to compute reasoning vector.

Method	GSM8K		MATH 500		AIME 2024		AVG	
	ACC ↑	LEN ↑	ACC ↑	LEN ↑	ACC ↑	LEN ↓	ACC ↑	LEN ↓
different α								
Original	77.9	955	75.4	3662	20.8	7333	-	-
ERRV-Cosine	79.0	720	79.6	2175	22.1	4888	2.20	-32.85%
ERRV-Constant	73.4	614	63.2	1436	10.4	1641	-9.03	-58.04%
ERRV-Dynamic	79.0	710	79.8	2280	21.8	4826	+2.17	-32.53%
ERRV-IS-Cosine	82.1	824	83.0	2598	28.5	6426	+6.50	-18.38%
ERRV-IS-Constant	82.1	880	82.8	2610	28.8	6415	+6.53	-16.37%
ERRV-IS-Dynamic	82.3	875	82.5	2612	28.8	6430	+6.50	-16.45%
different n								
ERRV-n10	79.0	720	79.6	2175	22.1	4888	2.20	-32.85%
ERRV-n20	79.1	722	79.8	2180	22.3	4850	+2.37	-32.91%
ERRV-n200	79.2	718	80.2	2070	23.1	4844	+2.80	-34.08%
ERRV-IS-n10	82.1	824	83.0	2598	28.5	6426	+6.50	-18.38%
ERRV-IS-n20	82.0	880	83.0	2490	28.8	6430	+6.57	-17.39%
ERRV-IS-n200	82.6	850	83.2	2450	29.0	6390	+6.9	-18.98%

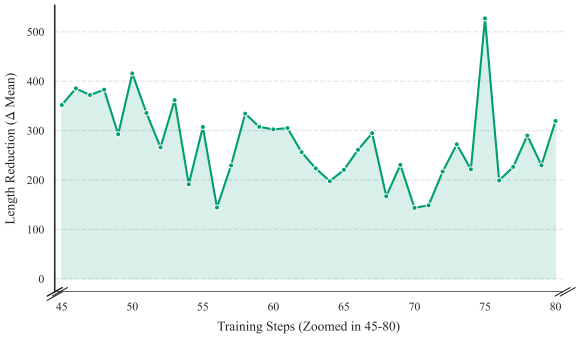


Figure 7: Average length difference between unsteered and steered rollouts on DS-1.5B during ERRV training.