STABILITY MAKES CONFIDENCE RELIABLE: TEST-TIME ADAPTATION FOR LLM REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Test-time adaptation has emerged as a lightweight alternative to costly post-training for improving the reasoning capabilities of Large Language Models (LLMs) on downstream tasks. Predictive entropy provides a model-derived signal for such adaptation, guiding models toward higher-confidence reasoning states without external verifiers or reward models. However, higher confidence does not necessarily imply correctness, as LLMs may remain highly confident along incorrect reasoning trajectories. We observe that high-confidence reasoning is more likely to be correct when confidence remains stable under local perturbations. Based on this observation, we propose Test-Time Adaptation via Stability-Aware Confidence Optimization (TASCO), a framework that incorporates local stability into confidence-based test-time adaptation while keeping the LLM frozen. TASCO operationalizes local stability by optimizing a lightweight task-level prefix under two alternative perturbation strategies: Random Perturbation promotes distributional stability across trajectories induced by nearby perturbed prefixes, whereas Sharpness-Aware Perturbation targets worst-case local sensitivity. Experiments demonstrate that TASCO improves reasoning accuracy and token efficiency across diverse LLMs and reasoning benchmarks, while behavioral analyses show that it maintains stable confidence under local perturbations without prematurely concentrating the model's predictive distribution.

1 INTRODUCTION

Large Language Models (LLMs) have demonstrated strong reasoning capabilities on complex tasks such as mathematical reasoning and code generation Guo et al. (2025); Jaech et al. (2024). Traditionally, adapting these models to downstream tasks relies on post-training methods that update model parameters, such as supervised fine-tuning or reinforcement learning Shao et al. (2024); Lambert et al. (2024). Although effective, these methods require substantial computational resources, motivating recent work to keep model parameters frozen and instead optimize continuous inputs or internal representations at inference time Hu et al. (2025).

One effective paradigm is to optimize a small set of continuous variables, including soft prompts, steering vectors, and latent representations, to guide a frozen LLM toward task-specific reasoning states Xu et al. (2025b); Li et al. (2025); Sinii et al. (2025). These approaches differ mainly in the source of their optimization signals. Some rely on external verifiers or reward models to score answers or reasoning trajectories Khalifa et al. (2025); Nguyen & Le (2026), whereas verifier-free methods use the model's output entropy as an intrinsic confidence signal Ye et al. (2025).

However, entropy-based test-time optimization assumes that an LLM is more likely to answer correctly when its predictive entropy is lower during reasoning Agarwal et al. (2026). This assumption can fail on complex reasoning tasks, where LLMs can be poorly calibrated and remain highly confident along incorrect reasoning trajectories Jurayj et al. (2025); Huang et al. (2025). When confidence is the only test-time objective, entropy minimization cannot distinguish high confidence associated with reliable reasoning from spurious confidence along an incorrect trajectory, potentially steering the model toward a locally fragile high-confidence state Chen et al. (2026a); Zhao et al. (2026).

How can we guide an LLM toward a high-confidence state that supports reliable reasoning? Inspired by the connection between flat minima and generalization Keskar et al. (2016), we examine the local stability of high-confidence states. Specifically, high confidence that remains stable under

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

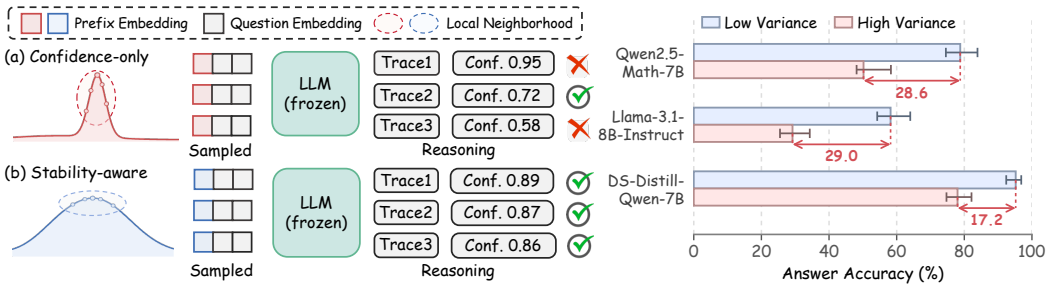


Figure 1: Confidence stability and answer accuracy. Under prefix perturbations, confidence-only optimization can yield an incorrect answer with the highest confidence, whereas the stability-aware case shows more consistent confidence and correct answers (left). Across LLMs from three families, answer accuracy is compared between low- and high-variance query groups (right).

local perturbations to the steering variable is more likely to stem from reliable reasoning than from a fragile overconfident state, as illustrated in Fig. 1 (left). We measure this stability using trajectory-level confidence variance. As shown in Fig. 1 (right), the low-variance group consistently achieves higher answer accuracy than the high-variance group across representative models from three families. Further analysis shows that the low-variance group remains more accurate among queries with comparable mean confidence (see Appendix B for details). Together, these results suggest that local stability complements confidence magnitude in characterizing reasoning reliability.

Guided by this observation, we propose Test-Time Adaptation via Stability-Aware Confidence Optimization (TASCO), a label-free framework for guiding frozen LLMs toward reliable high-confidence reasoning. TASCO optimizes a task-level prefix shared across unlabeled test queries to stabilize confidence during reasoning. Once optimized, this prefix is prepended to each query embedding, steering the frozen model throughout trace generation. Since local stability of reasoning confidence can manifest as either variation across nearby perturbed prefixes or sensitivity to the most adversely perturbed prefix, TASCO operationalizes it using two complementary strategies: Random Perturbation measures distributional stability through confidence variation across trajectories independently decoded under Gaussian-perturbed prefixes, while Sharpness-Aware Perturbation targets worst-case local sensitivity by applying a single gradient-guided perturbation along the most sensitive direction in the prefix space for the current trajectory. Optimizing the prefix with both schemes, TASCO improves accuracy by up to 17.2% while generating up to 28.1% fewer tokens across diverse LLMs and reasoning benchmarks. Behavioral analyses show TASCO promotes stable confidence under local perturbations without prematurely narrowing reasoning directions. The learned prefix improves reasoning on unseen datasets without further optimization.

Our contributions are summarized as follows:

- We introduce a stability-aware perspective on confidence-driven test-time optimization, identifying local stability as a complementary signal for assessing the reliability of high-confidence reasoning.
- We propose TASCO, a label-free framework that uses two alternative perturbation strategies to guide frozen LLMs toward high-confidence states that remain stable under local perturbations. We further characterize how both strategies control local prefix sensitivity.
- Extensive experiments across diverse LLMs and reasoning benchmarks demonstrate that TASCO improves reasoning accuracy and efficiency while promoting stable confidence under local perturbations.

2 Related Work

2.1 Reasoning Representations in LLMs

Chain-of-thought (CoT) prompting elicits explicit intermediate steps to improve LLM reasoning Wei et al. (2022). Subsequent work improves the generation or selection of these steps through trajectory

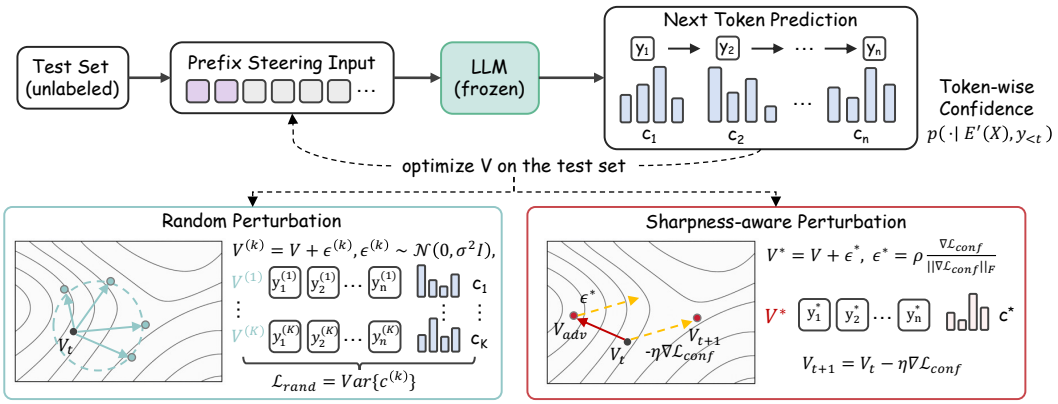


Figure 2: Overview of TASCO. TASCO optimizes a shared prefix for confidence and stability. Random Perturbation penalizes local confidence variance, while Sharpness-Aware Perturbation targets worst-case sensitivity.

aggregation Wang et al. (2022), structured search Yao et al. (2023), and step-level verification Zhang et al. (2025b). Reasoning-enhanced models extend this direction by generating longer deliberative traces Chen et al. (2026b). Continuous reasoning takes a different approach: COCONUT and CODI train models to express intermediate computation in latent states Hao et al. (2024); Shen et al. (2025), while SoftCoT and SemCoT use learned auxiliary components to supply implicit reasoning tokens Xu et al. (2025b); He et al. (2026). TASCO does not replace textual reasoning with latent thoughts. It keeps the backbone frozen and optimizes a task-level continuous prefix at test time to guide the reasoning trajectory the model subsequently generates.

2.2 Test-Time Adaptation for Reasoning

Test-time adaptation also differs in what it updates and which signals guide the update. TTRL uses majority-vote rewards from sampled answers to update model parameters Zuo et al. (2026), while other methods use entropy-derived objectives for parameter optimization Prabhudesai et al. (2025); Gao et al. (2025). Methods that leave the backbone frozen instead steer its activations or optimize continuous inputs Han et al. (2026); Chen et al. (2025). Among these, TTSV and LTPO are particularly close to TASCO. TTSV minimizes output entropy by optimizing an input-side steering vector shared across test queries Kang et al. (2026); LTPO optimizes latent thought vectors for individual queries using a confidence-based reward Ye et al. (2025). Both use model-derived confidence to guide adaptation, but neither explicitly constrains how confidence changes around the optimized continuous variables. TASCO retains confidence as a useful signal while also optimizing the local stability of trajectory-level confidence under prefix perturbations.

Perturbation sensitivity has also been studied as a diagnostic signal. CCPS uses changes in hidden representations to predict answer correctness Khanmohammadi et al. (2025), and Wen et al. perturb token embeddings to identify uncertain reasoning steps Wen et al. (2026). These methods assess a generated trajectory or its intermediate steps. TASCO instead uses local perturbations during adaptation, directly shaping the task-level prefix toward reasoning trajectories whose confidence is less sensitive to nearby changes.

3 Method

In this section, we present TASCO, a stability-aware framework that optimizes a continuous prefix at test time while keeping the LLM frozen. We first formulate the optimization problem and define the confidence objective. We then introduce two alternative strategies for characterizing local stability from distributional and worst-case perspectives. We further analyze how these strategies guide confidence optimization. Figure 2 provides an overview of TASCO.

3.1 PRELIMINARIES

Problem Formulation. Let M_θ denote a pretrained LLM with frozen parameters θ, and let $\mathcal{D}_{\text{test}} = \{X_1, \ldots, X_N\}$ be an unlabeled test set. Our goal is to improve the reasoning performance of M_θ on $\mathcal{D}_{\text{test}}$ without updating its parameters. To this end, we optimize a shared continuous prefix $V \in \mathbb{R}^{L \times d}$. Here, L and d denote the prefix length and embedding dimension. Given a query X with token embeddings $E(X)$, we prepend V to form the prefix-conditioned input $E'(X) = [V; E(X)]$, using $[\cdot;\cdot]$ to denote concatenation along the sequence dimension. The frozen model then autoregressively generates a reasoning trajectory $Y = (y_1, \ldots, y_T)$. At decoding step t, its next-token distribution is $p_\theta(\cdot \mid E'(X), y_{<t})$. TASCO optimizes V using only label-free signals derived from M_θ, without external supervision or auxiliary models.

Confidence Objective. We use predictive entropy as a label-free confidence signal. For a generated trajectory $Y = (y_1, \ldots, y_T)$, the confidence loss is defined as the average token-level entropy:

$$\mathcal{L}_{\text{conf}}(V; X, Y) = \frac{1}{T} \sum_{t=1}^{T} H\left(p_\theta(\cdot \mid E'(X), y_{<t})\right), \tag{1}$$

where $H(p) = -\sum_{w \in \mathcal{V}} p(w) \log p(w)$ denotes the entropy over the vocabulary $\mathcal{V}$. In practice, we compute this loss using one generated trajectory per input and treat the generated tokens as fixed.

3.2 CONFIDENCE OPTIMIZATION WITH LOCAL STABILITY

Although entropy minimization offers a label-free objective for prefix optimization, it may steer the prefix toward a locally fragile high-confidence state. TASCO therefore introduces two local-stability strategies: (1) Random Perturbation for distributional stability and (2) Sharpness-Aware Perturbation for worst-case directional stability.

3.2.1 RANDOM PERTURBATION.

We promote distributional stability by examining reasoning trajectories decoded under nearby prefixes. At each optimization step, we draw K Gaussian perturbations around the current prefix:

$$V^{(k)} = V + \epsilon^{(k)}, \quad \epsilon^{(k)} \sim \mathcal{N}(0, \sigma^2 I), \tag{2}$$

where σ is the standard deviation of the Gaussian noise applied to each prefix element. For each input X, the model greedily decodes a trajectory $Y^{(k)}$ under every perturbed prefix $V^{(k)}$. We summarize each rollout by the average log-likelihood of its generated tokens:

$$c^{(k)} = \frac{1}{T_k} \sum_{t=1}^{T_k} \log p_\theta\left(y_t^{(k)} \mid E'^{(k)}(X), y_{<t}^{(k)}\right). \tag{3}$$

Variation in $\{c^{(k)}\}_{k=1}^{K}$ reflects local behavioral instability and may capture trajectory switching because each perturbed prefix is re-decoded. We quantify the behavioral sharpness of V on X by the sample variance:

$$\mathcal{S}(V; X) = \frac{1}{K-1} \sum_{k=1}^{K} \left(c^{(k)} - \bar{c}\right)^2, \quad \bar{c} = \frac{1}{K} \sum_{k=1}^{K} c^{(k)}. \tag{4}$$

For a mini-batch $\mathcal{B}$, the random-perturbation regularizer is obtained by averaging this quantity across inputs, $\mathcal{L}_{\text{rand}}(V) = |\mathcal{B}|^{-1} \sum_{X \in \mathcal{B}} \mathcal{S}(V; X)$. Combining it with the batch-averaged confidence loss gives:

$$\mathcal{L}_{\text{RP}} = \mathcal{L}_{\text{conf}}(V) + \lambda_{\text{rand}} \mathcal{L}_{\text{rand}}(V), \tag{5}$$

where λ_{rand} balances predictive confidence and local stability. Thus, $\mathcal{L}_{\text{conf}}$ encourages low predictive entropy, whereas $\mathcal{L}_{\text{rand}}$ reduces behavioral variation across nearby prefixes.

3.2.2 SHARPNESS-AWARE PERTURBATION.

Unlike Random Perturbation, Sharpness-Aware Perturbation avoids generating multiple trajectories by constructing a single gradient-guided perturbation. At each update, we fix the trajectory Y decoded under V while minimizing the worst-case confidence loss:

$$\min_{V} \max_{\|\epsilon\|_F \le \rho} \mathcal{L}_{\text{conf}}(V + \epsilon; X, Y), \tag{6}$$

where ρ defines the perturbation radius under the Frobenius norm. Since the inner maximization in Eq. 6 is generally intractable, we use a first-order approximation of the confidence loss around V, based on

$$g = \nabla_V \mathcal{L}_{\text{conf}}(V; X, Y). \tag{7}$$

The linearized inner problem yields the following worst-case perturbation and perturbed prefix:

$$\widehat{V} = V + \epsilon^\star, \quad \epsilon^\star = \rho \frac{g}{\|g\|_F}, \tag{8}$$

where $\epsilon^\star$ is the perturbation along the steepest ascent direction under $\|\epsilon\|_F \le \rho$. We then evaluate the confidence loss at $\widehat{V}$ using the same trajectory Y:

$$\mathcal{L}_{\text{SAP}} = \mathcal{L}_{\text{conf}}(\widehat{V}; X, Y). \tag{9}$$

When updating V with $\mathcal{L}_{\text{SAP}}$, we treat $\epsilon^\star$ as constant to avoid second-order derivatives through g.

The variants trade off neighborhood coverage and computation: Random Perturbation captures variation across sampled perturbations, whereas SAP targets worst-case sensitivity with one gradient-guided perturbation. Complete algorithms appear in Appendix D.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Models. We evaluate TASCO on five open-source LLMs covering both general and reasoning models. General LLMs include Qwen2.5-Math (Yang et al., 2024) in two sizes (1.5B and 7B) and LLaMA-3.1-8B-Instruct (Grattafiori et al., 2024). Reasoning LLMs include DeepSeek-R1-Distill-Qwen in two sizes (1.5B and 7B) (Guo et al., 2025).

Datasets. We evaluate on six reasoning benchmarks. For mathematics, we use MATH-500 (Hendrycks et al., 2021), AMC23, AIME24 (Li et al., 2024), and AIME25 (Balunovic et al., 2026), spanning high-school to olympiad-level problems. For science, we use Minerva Math (Lewkowycz et al., 2022), with undergraduate-level STEM problems, and GPQA Diamond (Rein et al., 2023), with graduate-level questions written by domain experts.

Baselines. For general LLMs, we include CoT (Wei et al., 2022) as the standard prompting baseline. We compare with SLOT (Hu et al., 2025) and LatentSeek (Li et al., 2025), which perform test-time adaptation through final-layer or latent representations. We also compare with LTPO (Ye et al., 2025) and TTSV (Kang et al., 2026), which optimize continuous latent variables using model-internal confidence signals while keeping the backbone frozen. For reasoning LLMs, we compare with s1 (Muennighoff et al., 2025), CoD (Xu et al., 2025a), and α1 (Zhang et al., 2025a), which control reasoning behavior at test time.

Implementation Details. For general LLMs, we use a shared prefix of length $L = 20$ and a maximum generation length of 3,072 tokens. For reasoning-enhanced LLMs, we use $L = 10$ and allow up to 8,192 tokens for longer reasoning traces. Random Perturbation uses $K = 8$ perturbed rollouts with $\sigma = 0.01$, while Sharpness-Aware Perturbation uses a perturbation radius of $\rho = 0.8$. We further analyze sensitivity to σ and ρ. All experiments run on a single NVIDIA A800 GPU. More details are provided in Appendix A.

Evaluation. We report accuracy averaged over eight independent generations per problem on all benchmarks. For math benchmarks, we extract and score final answers using dataset-specific normalization and equivalence rules.

Table 1: Main performance comparison across three models and five reasoning benchmarks (accuracy, %). Δ denotes the absolute percentage-point change relative to Zero-Shot CoT. Best results are in **bold**; second-best results are underlined.

Model	Method	MATH500	AMC23	Minerva	AIME24	GPQA	Avg.	Δ
Qwen2.5-Math-1.5B	Zero-Shot CoT	39.4	34.7	8.8	6.7	20.2	22.0	–
	SLOT	43.4	47.2	16.9	10.0	26.8	28.9	+6.9
	LatentSeek	45.6	43.4	14.3	7.5	22.2	26.6	+4.6
	LTPO	58.6	45.6	12.9	9.2	21.7	29.6	+7.6
	TTSV	68.6	46.8	20.6	8.8	24.8	33.9	+11.9
	TASCO-RP	70.8	50.6	**22.4**	10.4	**27.8**	36.4	+14.4
	TASCO-SAP	**71.8**	**52.2**	21.3	**12.1**	26.8	**36.8**	+14.8
Qwen2.5-Math-7B	Zero-Shot CoT	52.2	44.4	12.9	14.6	31.8	31.2	–
	SLOT	58.8	56.2	29.4	18.8	32.8	39.2	+8.0
	LatentSeek	57.6	48.1	20.5	6.2	32.8	33.0	+1.8
	LTPO	65.6	42.8	17.6	18.8	33.5	35.7	+4.5
	TTSV	71.0	59.1	33.5	17.9	36.4	43.6	+12.4
	TASCO-RP	**75.8**	**68.8**	37.1	21.2	37.4	48.1	+16.9
	TASCO-SAP	73.4	66.2	**39.1**	**24.2**	**38.9**	**48.4**	+17.2
Llama3.1-8B-Instruct	Zero-Shot CoT	47.4	24.7	22.1	7.5	28.3	26.0	–
	SLOT	48.8	23.8	22.8	9.2	**33.8**	27.7	+1.7
	LatentSeek	49.4	26.8	23.2	3.3	32.3	27.0	+1.0
	LTPO	48.4	27.5	20.5	11.3	30.5	27.6	+1.6
	TTSV	47.8	22.8	22.1	6.2	27.8	25.3	-0.7
	TASCO-RP	46.8	**29.1**	25.0	10.4	33.3	28.9	+2.9
	TASCO-SAP	**50.2**	28.4	**26.5**	**11.7**	31.8	**29.7**	+3.7

4.2 Main Results

Tables 1 and 2 compare three general and two reasoning-enhanced LLMs. TASCO-SAP achieves the highest average accuracy across all five backbones, raising scores on general LLMs from 22.0% to 36.8% (Qwen2.5-Math-1.5B), 31.2% to 48.4% (Qwen2.5-Math-7B), and 26.0% to 29.7% (Llama3.1-8B-Instruct), with gains across model families and scales.

Effectiveness of stability-aware confidence optimization. TASCO-SAP attains higher average accuracy than confidence-guided test-time adaptation baselines on three general LLMs. TTSV optimizes input-side steering vectors, whereas LTPO optimizes latent thought vectors. On Qwen2.5-Math-1.5B/7B and Llama3.1-8B-Instruct, TASCO-SAP exceeds TTSV by 2.9, 4.8, and 4.4 points and LTPO by 7.2, 12.7, and 2.1 points, respectively. On models with long reasoning traces, directly applying TTSV and LTPO reduces accuracy, whereas TASCO-SAP raises it from 45.8% to 52.6% on DeepSeek-R1-Distill-Qwen-1.5B and from 59.5% to 65.5% on DeepSeek-R1-Distill-Qwen-7B. TASCO-SAP also achieves the highest average accuracy among the compared methods and tops three of four benchmarks per backbone.

Improved Token Efficiency. Figure 3 shows that TASCO-RP and TASCO-SAP reduce average generation length by 28.1% and 24.0%, respectively, relative to Zero-Shot CoT across general LLMs while improving accuracy. On reasoning-enhanced models, Table 2 shows that TASCO-SAP reduces average length from 6299 to 5372 tokens (14.7%) on the 1.5B backbone and from 5517 to 5042 tokens (8.6%) on the 7B backbone. TASCO thus achieves higher accuracy with shorter outputs across both model types.

4.3 Ablation and Sensitivity Analyses

Complementarity of confidence and stability. We compare TASCO with four controlled variants on MATH500 and Minerva using Qwen2.5-Math-7B and Llama3.1-8B-Instruct. CoT-Unk adds unoptimized prefix positions; Confidence Only optimizes confidence alone; Perturbed Confidence uses the same K perturbed rollouts as TASCO-RP without the stability objective; and Stability Only retains the stability objective without confidence optimization.

Table 2: Accuracy (%) and average generation length (tokens) of reasoning-enhanced LLMs. Best results are in **bold**; second-best results are underlined.

Method	AIME24		AIME25		MATH500		AMC23		Average	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
DeepSeek-R1-Distill-Qwen-1.5B										
Zero-Shot-CoT	22.8	7580	22.5	7674	78.6	3700	59.4	6242	45.8	6299
s1	25.8	7778	25.4	7982	80.2	4623	61.9	6428	48.3$_{\uparrow 2.5}$	6703$_{\uparrow 6.4\%}$
CoD	26.7	7196	20.8	7552	79.4	**3267**	66.2	5293	48.3$_{\uparrow 2.5}$	5827$_{\downarrow 7.5\%}$
α1	**30.8**	6841	12.9	**6529**	81.0	3541	63.1	4310	47.0$_{\uparrow 1.2}$	**5305**$_{\downarrow 15.8\%}$
TASCO-SAP	30.4	**6751**	**29.2**	6838	**81.4**	3616	**69.4**	**4282**	**52.6**$_{\uparrow 6.8}$	5372$_{\downarrow 14.7\%}$
DeepSeek-R1-Distill-Qwen-7B										
Zero-Shot-CoT	44.2	6899	25.0	7099	89.4	3399	79.4	4672	59.5	5517
s1	37.5	7368	24.2	7041	89.0	4072	78.1	5271	57.2$_{\downarrow 2.3}$	5938$_{\uparrow 7.6\%}$
CoD	45.0	6301	33.3	**6582**	88.0	**2071**	85.6	**3645**	63.0$_{\uparrow 3.5}$	**4650**$_{\downarrow 15.7\%}$
α1	36.7	6831	32.5	6638	89.8	3848	**90.6**	4529	62.4$_{\uparrow 2.9}$	5462$_{\downarrow 1.0\%}$
TASCO-SAP	**49.2**	**6188**	**34.2**	6652	**90.4**	3041	88.1	4287	**65.5**$_{\uparrow 6.0}$	5042$_{\downarrow 8.6\%}$

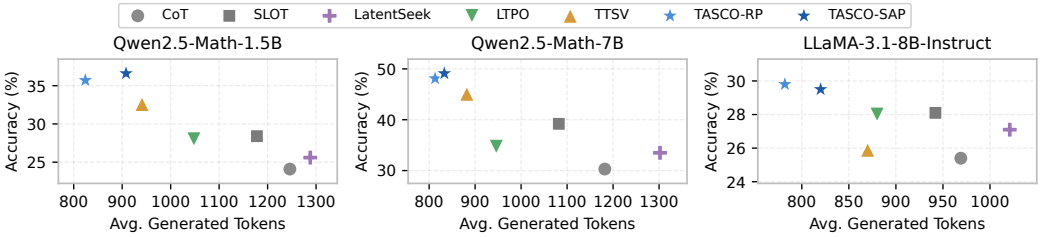


Figure 3: Accuracy-token trade-off across general LLMs, averaged over all benchmarks.

As shown in Figure 4, CoT-Unk improves some results but does not reproduce TASCO's gains, so extra prefix positions alone are insufficient. Optimizing those positions for confidence also gives inconsistent accuracy gains. The comparison with Perturbed Confidence then isolates the role of local stability from that of additional rollouts: TASCO-RP performs better despite using the same number of perturbed trajectories. Conversely, Stability Only falls short of the best TASCO variant, indicating that reducing variability also needs confidence guidance. Taken together, these controls support complementary roles for the two objectives: confidence defines the optimization target, while local stability discourages solutions whose confidence changes sharply under nearby prefix perturbations.

Sensitivity to perturbation scale. On Qwen2.5-Math-1.5B, we vary TASCO-RP's Gaussian scale σ and TASCO-SAP's radius ρ across AMC23, AIME24, Minerva, and GPQA, fixing other settings. Figure 5 shows both variants outperform the zero-perturbation confidence-only reference at every tested nonzero scale. TASCO-RP peaks at $\sigma = 0.01$ on all four datasets, though intermediate changes can be nonmonotonic. TASCO-SAP peaks at $\rho = 0.8$ on AMC23 and AIME24 and $\rho = 0.4$ on Minerva and GPQA; its default $\rho = 0.8$ still beats the reference on all four. Gains persist across scales, though optimal scales vary by dataset and larger perturbations can reduce accuracy.

4.4 Effects on Reasoning Behavior

Robustness to local prefix perturbations. We first examine whether the learned prefixes preserve their behavior under small local perturbations on Qwen2.5-Math-7B. For each input, we add Gaussian noise with σ=0.01 to the learned prefix and generate K=8 perturbed rollouts. We measure trajectory-confidence variance and answer consistency to capture variation in confidence scores and final answers.

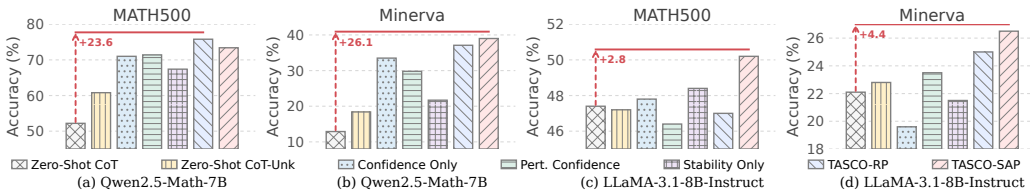


Figure 4: Impact of optimized prefixes on MATH500 and Minerva. The results indicate that the gains of TASCO come from learning stability-aware task-level prefixes rather than simply adding extra prefix positions.

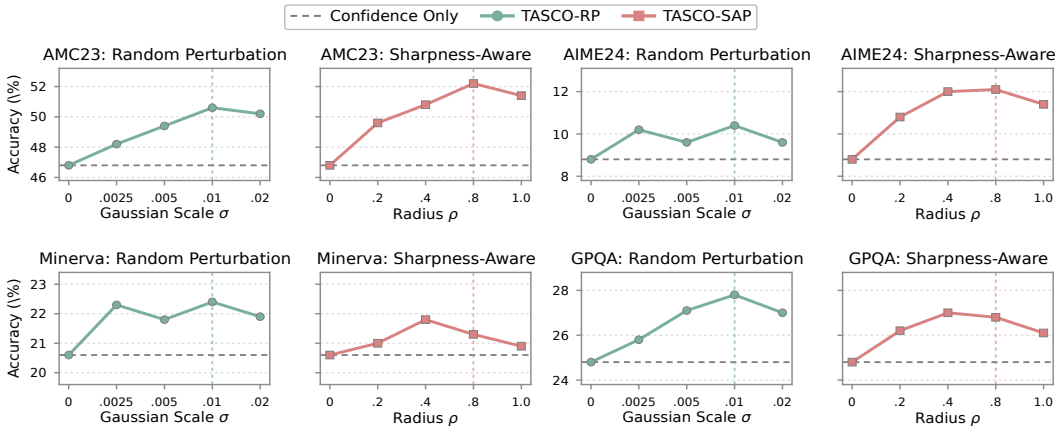


Figure 5: Sensitivity of TASCO to the perturbation scale on Qwen2.5-Math-1.5B.

Table 3 shows that both TASCO variants have lower trajectory-confidence variance and higher answer consistency than Confidence Only across all four datasets. On MATH500, TASCO-RP and TASCO-SAP reduce variance from 4.83 to 2.33 and 2.89, respectively, while increasing consistency from 62.9% to 71.7% and 73.8%. The same pattern holds on other benchmarks. Lower variance indicates that confidence changes less under nearby prefix perturbations, while higher consistency shows that final answers change less often. Together, these results show that TASCO's prefixes support more consistent behavior under local perturbations, as intended by stability-aware optimization.

Table 3: Local stability under prefix perturbations on Qwen2.5-Math-7B.

Method	MATH500	AMC23	Minerva	GPQA
Confidence Variance ↓				
Confidence Only	4.83	3.05	6.59	15.76
TASCO-RP	2.33	2.34	5.48	9.31
TASCO-SAP	2.89	2.32	4.51	12.26
Answer Consistency ↑				
Confidence Only	62.9	42.5	58.9	48.1
TASCO-RP	71.7	47.0	62.6	59.4
TASCO-SAP	73.8	50.5	66.6	52.8

Reduced early confidence concentration. To examine whether TASCO reduces premature commitment to a high-probability token choice, we analyze next-token distributions over the first 5% of generation on MATH500 with Qwen2.5-Math-7B. Averaged across 500 queries, TASCO-RP and TASCO-SAP show higher entropy and varentropy and lower top-1 probability than Confidence Only at the start of decoding (Figure 6). Thus, more probability mass remains on alternative tokens early in generation instead of being concentrated on the top choice. Together with the accuracy gains, this less concentrated early distribution may help the model follow more reliable reasoning paths.

Performance Across Query Difficulty. To test whether TASCO benefits queries the unadapted model answers less reliably, we compare TASCO-RP with Confidence Only across MATH500 difficulty groups. Appendix E.1 shows that TASCO-RP raises hard-query accuracy from 46.7% to 54.4% on Qwen2.5-Math-7B and from 9.7% to 14.4% on LLaMA-3.1-8B-Instruct. Medium-query accu-

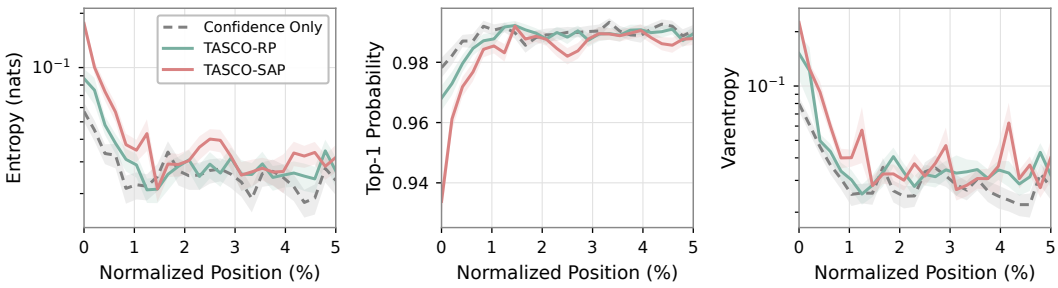


Figure 6: Early decoding dynamics on MATH500 with Qwen2.5-Math-7B. Mean next-token entropy, top-1 probability, and varentropy are shown over the first 5% of generated tokens.

racy rises by 5.4 percentage points for both models, while easy-query changes are small and mixed. The gains therefore extend beyond reliably solved queries.

4.5 Analysis of Generalization

Generalization Across Mathematical Tasks. We test whether a prefix optimized on one mathematical benchmark improves performance on others without re-optimization. As shown in Fig. 8 in the appendix, transferred prefixes outperform Zero-Shot CoT across the evaluated source–target pairs. These gains suggest that TASCO guides mathematical reasoning across benchmarks, rather than improving performance only on the source distribution.

Generalization to Other Domains. To examine whether TASCO applies beyond mathematical reasoning, we evaluate it on StrategyQA Geva et al. (2021) for commonsense reasoning and the STEM subset of MMLU-Pro Wang et al. (2024) for multidisciplinary knowledge and reasoning. We optimize a separate prefix for each benchmark while keeping the backbone model frozen.

Table 4 shows that TASCO achieves the highest accuracy on both benchmarks for each backbone. Relative to Zero-Shot CoT, its average accuracy increases from 54.06% to 56.33% on LLaMA-3.1-8B-Instruct and from 53.65% to 57.49% on Qwen2.5-Math-7B. The gains extend to both task types: on MMLU-Pro STEM, accuracy rises by 2.79 and 3.38 percentage points, respectively, while on StrategyQA it rises by 1.75 and 4.29 points. LTPO and TTSV yield smaller gains or reduce the average accuracy in these settings. These results indicate that stability-aware prefix optimization is also effective when applied separately to commonsense and multidisciplinary reasoning tasks.

Table 4: Accuracy (%) on StrategyQA and MMLU-Pro STEM. Best results are in **bold**; second-best results are underlined.

Model	Method	Strategy-QA	MMLUPro-STEM	Avg.
LLaMA-3.1-8B-Instruct	Zero-Shot CoT	71.49	36.62	54.06
	LTPO	<u>72.18</u>	<u>38.14</u>	<u>55.16</u>
	TTSV	67.34	37.22	52.28
	TASCO (Ours)	**73.24**	**39.41**	**56.33**
Qwen2.5-Math-7B	Zero-Shot CoT	61.22	46.08	53.65
	LTPO	60.20	47.22	53.71
	TTSV	<u>64.08</u>	<u>48.40</u>	<u>56.24</u>
	TASCO (Ours)	**65.51**	**49.46**	**57.49**

5 Conclusion

We observed that high-confidence reasoning is more likely to yield correct answers when it remains stable under local perturbations. Based on this observation, we proposed TASCO, a label-free framework that optimizes a shared task-level prefix for confidence and stability while keeping the LLM frozen. Random Perturbation captures distributional variation across nearby behaviors, whereas Sharpness-Aware Perturbation targets worst-case local sensitivity. Experiments across diverse LLMs and reasoning benchmarks show that stability-aware confidence optimization improves reasoning accuracy and generation efficiency without parameter updates or external supervision.

AI USE STATEMENT

Generative AI tools, including OpenAI Codex, assisted with literature discovery, translation, manuscript editing, figure refinement, experimental design and feedback, evaluation code, and result interpretation. They also assisted with formulating and presenting the allocation bound and its proof. Reported measurements are based on model executions and recorded evaluation outputs, not generated estimates. The authors are responsible for the final text, code, claims, and results.

REFERENCES

Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. The unreasonable effectiveness of entropy minimization in llm reasoning. *Advances in Neural Information Processing Systems*, 38:107150–107180, 2026.

Mislav Balunovic, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Matharena: Evaluating llms on uncontaminated math competitions. *Advances in Neural Information Processing Systems*, 38, 2026.

Guohao Chen, Shuaicheng Niu, Chen Deyu, Jiahao Yang, Zitian Zhang, Mingkui Tan, Pengcheng Wu, and Zhiqi Shen. Zerosiam: An efficient asymmetry for test-time entropy optimization without collapse. In *International Conference on Learning Representations*, volume 2026, pp. 144893–144925, 2026a.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *Science China Information Sciences*, 69(6):161101, 2026b.

Runjin Chen, Zhenyu Zhang, Junyuan Hong, Souvik Kundu, and Zhangyang Wang. Seal: Steerable reasoning calibration of large language models for free. *arXiv preprint arXiv:2504.07986*, 2025.

Zitian Gao, Lynx Chen, Haoming Luo, Joey Zhou, and Bryan Dai. One-shot entropy minimization. *arXiv preprint arXiv:2505.20282*, 2025.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021.

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

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Pengrui Han, Xueqiang Xu, Keyang Xuan, Peiyang Song, Siru Ouyang, Runchu Tian, Yuqing Jiang, Cheng Qian, Pengcheng Jiang, Jiashuo Sun, et al. Steer2adapt: Dynamically composing steering vectors elicits efficient adaptation of llms. *arXiv preprint arXiv:2602.07276*, 2026.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

Yinhan He, Wendy Zheng, Yaochen Zhu, Zaiyi Zheng, Lin Su, Sriram Vasudevan, Qi Guo, Liangjie Hong, and Jundong Li. Semcot: Accelerating chain-of-thought reasoning through semantically-aligned implicit tokens. *Advances in Neural Information Processing Systems*, 38:43455–43485, 2026.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Yang Hu, Xingyu Zhang, Xueji Fang, Zhiyang Chen, Xiao Wang, Huatian Zhang, and Guojun Qi. Slot: Sample-specific language model optimization at test-time. *arXiv preprint arXiv:2505.12392*, 2025.

Chengsong Huang, Langlin Huang, Jixuan Leng, Jiacheng Liu, and Jiaxin Huang. Efficient test-time scaling via self-calibration. *arXiv preprint arXiv:2503.00031*, 2025.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

William Jurayj, Jeffrey Cheng, and Benjamin Van Durme. Is that your final answer? test-time scaling improves selective question answering. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 636–644, 2025.

Xinyue Kang, Diwei Shi, and Li Chen. Model whisper: Steering vectors unlock large language models' potential in test-time. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 31392–31400, 2026.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

Muhammad Khalifa, Rishabh Agarwal, Lajanugen Logeswaran, Jaekyeom Kim, Hao Peng, Moontae Lee, Honglak Lee, and Lu Wang. Process reward models that think. *arXiv preprint arXiv:2504.16828*, 2025.

Reza Khanmohammadi, Erfan Miahi, Mehrsa Mardikoraem, Simerjot Kaur, Ivan Brugere, Charese Smiley, Kundan S Thind, and Mohammad M Ghassemi. Calibrating llm confidence by probing perturbed representation stability. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 10459–10525, 2025.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

Hengli Li, Chenxi Li, Tong Wu, Xuekai Zhu, Yuxuan Wang, Zhaoxin Yu, Eric Hanchen Jiang, Song-Chun Zhu, Zixia Jia, Ying Nian Wu, et al. Seek in the dark: Reasoning via test-time instance-level policy gradient in latent space. *arXiv preprint arXiv:2505.13308*, 2025.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, et al. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9, 2024.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20286–20332, 2025.

Tuc Nguyen and Thai Le. Atlas: Adaptive test-time latent steering with external verifiers for enhancing llms reasoning. *arXiv preprint arXiv:2601.03093*, 2026.

Mihir Prabhudesai, Lili Chen, Alex Ippoliti, Katerina Fragkiadaki, Hao Liu, and Deepak Pathak. Maximizing confidence alone improves reasoning. *arXiv preprint arXiv:2505.22660*, 2025.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. Codi: Compressing chain-of-thought into continuous space via self-distillation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 677–693, 2025.

Viacheslav Sinii, Alexey Gorbatovski, Artem Cherepanov, Boris Shaposhnikov, Nikita Balagansky, and Daniil Gavrilov. Steering llm reasoning through bias-only adaptation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 9213–9222, 2025.

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

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290, 2024.

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

Qihao Wen, Jiahao Wang, Yang Nan, Pengfei He, Ravi Tandon, and Han Xu. Embedding perturbation may better reflect the uncertainty in llm reasoning. *arXiv preprint arXiv:2602.02427*, 2026.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. Chain of draft: Thinking faster by writing less. *arXiv preprint arXiv:2502.18600*, 2025a.

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

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Wengao Ye, Yan Liang, and Lianlei Shan. Thinking on the fly: Test-time reasoning enhancement via latent thought policy optimization. *arXiv preprint arXiv:2510.04182*, 2025.

Junyu Zhang, Runpei Dong, Han Wang, Xuying Ning, Haoran Geng, Peihao Li, Xialin He, Yutong Bai, Jitendra Malik, Saurabh Gupta, et al. Alphaone: Reasoning models thinking slow and fast at test time. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 11340–11365, 2025a.

Yufeng Zhang, Xuepeng Wang, Lingxiang Wu, and Jinqiao Wang. Enhancing chain of thought prompting in large language models via reasoning patterns. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 25985–25993, 2025b.

Chu Zhao, Enneng Yang, Yuting Liu, Jianzhe Zhao, and Guibing Guo. Echo: Entropy-confidence hybrid optimization for test-time reinforcement learning. *arXiv preprint arXiv:2602.02150*, 2026.

Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Xinwei Long, Ermo Hua, Biqing Qi, et al. Ttrl: Test-time reinforcement learning. *Advances in Neural Information Processing Systems*, 38:131459–131483, 2026.

A Experimental Details

This section provides the implementation details of TASCO and the compared baselines. We first describe the prefix optimization and perturbation procedures, followed by the baseline implementations and evaluation protocol.

Table 5: Model-specific optimization and generation settings. "Conf. tokens" denotes the maximum number of generated tokens used to compute the confidence objective.

Model family	Prefix length	Batch size	Learning rate	Temp.	Top-p	Rep. penalty	Conf. tokens	Max. tokens
Qwen2.5-Math	20	16	1e-3	0.7	0.95	1.15	256	3,072
LLaMA-3.1-Instruct	20	16	5e-6	0.7	0.95	1.05	256	3,072
DeepSeek-R1-Distill-Qwen	10	16	5e-4	0.6	0.95	1.00	1,024	8,192

Optimization and Generation Settings. We optimize only the continuous prefix V using AdamW while keeping all parameters of M_θ frozen. During optimization and evaluation, we generate trajectories using the model-specific temperature, top-p, and repetition penalty in Table 5. The confidence objective uses at most the first T_{opt} generated tokens, listed as "Conf. tokens" in the table. For final evaluation, we sample eight responses per problem with independent random seeds, subject to the corresponding maximum generation length.

Prefix Initialization. We use model-family-specific initialization. Prefixes for Qwen-based models, including DeepSeek-R1-Distill-Qwen, are sampled from $\mathcal{N}(0, I)$. For LLaMA models, we sample from a Gaussian distribution matching the empirical mean and variance of token embeddings in the unlabeled optimization set. We then perform a confidence-only warm-up and use the resulting prefix to initialize both TASCO variants.

Perturbation Settings. For Random Perturbation, we draw $K = 8$ Gaussian perturbations with $\sigma = 0.01$ at each optimization step. For Sharpness-Aware Perturbation, we use a perturbation radius of $\rho = 0.8$. The two variants are optimized and evaluated separately.

Trajectory Generation and Updates. At each optimization step, the model generates one trajectory for every input in the current mini-batch using the current prefix. The generated tokens are held fixed during backpropagation, and gradients are computed only with respect to V. New trajectories are generated after each prefix update. Random Perturbation re-decodes a trajectory under each perturbed prefix, whereas SAP evaluates its gradient-guided perturbation using the trajectory fixed within the current update.

A.1 Baseline Implementation Details

We use the official implementations and follow the configurations reported by the original authors whenever available.

Chain-of-Thought. CoT (Wei et al., 2022) prompts the model to produce explicit natural-language reasoning steps before its final answer. It is a prompting-only baseline: the model parameters and internal representations are not optimized.

SLOT. SLOT (Hu et al., 2025) performs test-time adaptation separately for each query by optimizing a sample-specific vector with the cross-entropy loss on the input prompt. The vector is added to the final hidden representations before the output layer and held fixed during answer generation, while the LLM remains frozen.

LatentSeek. LatentSeek (Li et al., 2025) optimizes instance-specific latent reasoning representations at test time while keeping the LLM parameters fixed. It uses policy-gradient updates guided by self-generated rewards to iteratively refine the latent representations.

LTPO. LTPO (Ye et al., 2025) applies online policy-gradient optimization to intermediate latent thought vectors without updating the LLM parameters. Its intrinsic confidence reward is computed from the frozen model's output distributions, requiring no external supervision.

TTSV. TTSV (Kang et al., 2026) learns a task-level steering vector from the unlabeled test set while keeping the LLM frozen. The vector is prepended to the input and optimized to reduce predictive entropy, steering the model toward higher-confidence outputs.

s1. s1 (Muennighoff et al., 2025) uses budget forcing to control the model's reasoning length at inference time. When the model attempts to end its reasoning, additional `Wait` tokens can prompt it to continue thinking.

Chain of Draft. CoD (Xu et al., 2025a) prompts the model to write concise intermediate reasoning steps that retain only essential information. Its standard prompt limits each draft step to a few words, reducing verbosity while preserving a step-by-step reasoning format.

α1. α1 (Zhang et al., 2025a) controls the transition from slow to fast reasoning at test time. Before the α-moment, it schedules reasoning-transition tokens dynamically; after that point, it terminates slow thinking to encourage faster answer generation.

Prompt Design. We use fixed prompt templates throughout optimization and evaluation. The prompt templates for each model family and task format are provided in Table 11.

B Exploring Local Confidence Stability

We investigate whether local confidence stability is associated with answer correctness and whether this relationship persists after accounting for confidence magnitude and query difficulty. We first establish the basic association and then examine these two alternative explanations.

B.1 Experimental Setup.

We conduct the analysis on the full MATH500 benchmark using five models from three families: Qwen2.5-Math-1.5B and 7B, Llama-3.1-8B-Instruct, and DeepSeek-R1-Distill-Qwen-1.5B and 7B. For each model, we use the final task-level prefix V obtained through confidence-only optimization. For every query X_i, we sample $K = 8$ Gaussian perturbations with $\sigma = 0.01$:

$$V^{(k)} = V + \epsilon^{(k)}, \qquad \epsilon^{(k)} \sim \mathcal{N}(0, \sigma^2 I). \tag{10}$$

Under each perturbed prefix, the model greedily decodes a trajectory $Y_i^{(k)}$. Its trajectory-level confidence is defined as the length-normalized log-probability:

$$c_i^{(k)} = \frac{1}{T_i^{(k)}} \sum_{t=1}^{T_i^{(k)}} \log p_\theta \left(y_{i,t}^{(k)} \mid X_i, V^{(k)}, y_{i,<t}^{(k)} \right), \tag{11}$$

where $y_{i,t}^{(k)}$ denotes the t-th token in $Y_i^{(k)}$ and $T_i^{(k)}$ is the total sequence length. We quantify local confidence stability using the sample variance:

$$\mathcal{S}_i = \frac{1}{K-1} \sum_{k=1}^{K} \left(c_i^{(k)} - \bar{c}_i \right)^2, \qquad \bar{c}_i = \frac{1}{K} \sum_{k=1}^{K} c_i^{(k)}, \tag{12}$$

where lower $\mathcal{S}_i$ indicates greater local stability, and $\bar{c}_i$ denotes the mean confidence across the perturbed trajectories.

For each query, we additionally greedily decode a reference trajectory under the unperturbed prefix V. We evaluate its final answer correctness using the standard benchmark evaluator and compute its original trajectory-level confidence $c_i^{(0)}$ following Eq. 11.



Figure 7: Answer accuracy under the unperturbed prefix for low- and high-variance query groups across five models. Groups are defined by the model-specific median of $\mathcal{S}_i$; error bars show 95% bootstrap confidence intervals, and red annotations indicate accuracy gaps.

Table 6: Association between local confidence variance and unperturbed correctness. Spearman's ρ measures the correlation between $\mathcal{S}_i$ and correctness, while AUROC uses $-\mathcal{S}_i$ to distinguish correct from incorrect answers.

Model	Spearman's ρ	AUROC
Qwen2.5-Math-1.5B	-0.369	0.668
Qwen2.5-Math-7B	-0.403	0.687
Llama-3.1-8B-Instruct	-0.413	0.683
DS-Distill-Qwen-1.5B	-0.308	0.709
DS-Distill-Qwen-7B	-0.359	0.745

B.2 Accuracy across Stability Groups

We first examine whether local confidence stability is associated with correctness at the unperturbed prefix. For each query, we estimate $\mathcal{S}_i$ from perturbed trajectories and independently evaluate correctness using a trajectory decoded under the unperturbed prefix. We split queries at the model-specific median of $\mathcal{S}_i$ and report group accuracy with 95% bootstrap confidence intervals.

As shown in Figure 7, Low-Variance queries achieve higher unperturbed accuracy across all five models, with gaps ranging from 17.2 to 29.0 percentage points. This pattern holds for both general and reasoning-enhanced LLMs. Table 6 further shows consistently negative correlations between local variance and unperturbed correctness, with AUROCs ranging from 0.668 to 0.745. Importantly, stability is estimated from perturbed trajectories, whereas correctness is evaluated independently at the unperturbed prefix. The observed separation is therefore not merely induced by measuring variance and accuracy on the same rollouts. Instead, it suggests that neighborhood stability contains information about the reliability of the optimized prefix itself across model families.

B.3 Controlling for Confidence Magnitude.

A remaining concern is that local variance may merely reflect confidence magnitude, as more confident queries may also be inherently more stable. To separate these effects, we divide queries into five equal-frequency strata according to their mean confidence across perturbed trajectories. Within each stratum, we split queries at the median of $\mathcal{S}_i$ and compare their perturbation accuracy, thereby reducing confidence differences between the Low- and High-Variance groups. We estimate confidence intervals using stratified bootstrap sampling and assess the accuracy gap with a stratified permutation test. We additionally compute the partial Spearman correlation between local variance and perturbation accuracy while controlling for mean confidence.

As shown in Table 7, Low-Variance queries achieve higher perturbation accuracy after confidence stratification across all five models. The differences are statistically significant on four models, with accuracy gaps ranging from 11.6 to 30.5 percentage points. Partial Spearman analysis yields the same pattern, showing significant negative correlations after controlling for mean confidence on these four models. These results suggest that local stability generally provides reliability information beyond confidence magnitude.

Table 7: Confidence-controlled association between local variance and perturbation accuracy. $\mathrm{Gap}_{\mathrm{strat}}$ is the low- minus high-variance accuracy after five-way stratification by mean confidence. Partial Spearman correlation controls for mean confidence continuously.

Model	$\mathrm{Gap}_{\mathrm{strat}}$	p_{perm}	ρ_{partial}	p_{partial}
Qwen2.5-Math-1.5B	5.4	0.086	-0.108	0.128
Qwen2.5-Math-7B	11.6	< 0.001	-0.242	< 0.001
Llama-3.1-8B	23.3	< 0.001	-0.283	< 0.001
DS-Distill-Qwen-1.5B	30.5	< 0.001	-0.560	< 0.001
DS-Distill-Qwen-7B	8.7	< 0.001	-0.297	< 0.001

Qwen2.5-Math-1.5B is an exception, showing a non-significant 5.4-point gap and a near-zero partial correlation ($\rho = -0.108$, $p = 0.128$). Its local variance is strongly correlated with mean confidence ($\rho = -0.717$), suggesting that variance provides little additional information beyond confidence magnitude for this model.

Overall, these analyses show a consistent association between local confidence stability and answer accuracy across model families. The relationship persists among queries with comparable mean confidence and within different difficulty groups, indicating that it cannot be explained solely by confidence magnitude or easier queries. These findings provide empirical motivation for incorporating local stability into confidence-based test-time optimization. The main experiments further examine whether optimizing this signal improves reasoning performance.

C Theoretical Analysis of Local Stability

This appendix examines how the two TASCO objectives constrain local confidence sensitivity. We first characterize the variation measured by Random Perturbation (RP), including changes associated with re-decoding. We then analyze how Sharpness-Aware Perturbation (SAP) approximates the worst-case confidence loss on a fixed trajectory. Finally, we compare their local interpretations and provide additional results at stationary points.

C.1 Notation and Scope

We fix an input X and identify the prefix $V \in \mathbb{R}^{L\times d}$ with a vector in $\mathbb{R}^m$, where $m = Ld$. Vector norms and inner products then coincide with the Frobenius norm and inner product of the prefix matrices. Hessians are taken with respect to these m vectorized coordinates.

The two variants use different scalar objectives. For RP, let $Z(\epsilon)$ be the trajectory greedily decoded under $V + \epsilon$, and let C be its average token log-probability, as defined in Eq. 3. Both the prefix and the decoded trajectory may change with ϵ. For a fixed trajectory Y, we instead write $f_Y(W)$ for its teacher-forced average token log-probability under a prefix W. For SAP, let $\ell(W) = \mathcal{L}_{\mathrm{conf}}(W; X, Y)$ denote the entropy loss with Y held fixed throughout the update. The SAP arguments also apply to a mini-batch average with all generated trajectories held fixed. We state the assumptions needed for each result below.

C.2 Random Perturbation: Distributional Stability

RP measures how trajectory-level confidence varies across nearby prefixes. Because each perturbed prefix is re-decoded, this variation can arise both within a fixed trajectory and across different trajectories.

Proposition 1 (Population confidence variance). Let $\epsilon^{(1)}, \ldots, \epsilon^{(K)}$ be independent draws from $\mathcal{N}(0, \sigma^2 I)$, with $K \geq 2$. Let $C_1, \ldots, C_K$ be the corresponding re-decoded trajectory confidences, and assume $\mathbb{E}[C^2] < \infty$. The sample variance $\mathcal{S}(V; X)$ in Eq. 4 is an unbiased estimator of the population variance:

$$\mathbb{E}[\mathcal{S}(V; X)] = \mathcal{S}_\sigma(V; X) := \mathrm{Var}_{\epsilon\sim\mathcal{N}(0,\sigma^2 I)}[C]. \tag{13}$$

Moreover, writing $Z = Z(\epsilon)$, we have

$$\mathcal{S}_\sigma(V; X) = \mathbb{E}_Z[\mathrm{Var}(C \mid Z)] + \mathrm{Var}_Z(\mathbb{E}[C \mid Z]). \tag{14}$$

Proof. Let $\mu = \mathbb{E}[C]$ and $\bar{C} = K^{-1}\sum_{k=1}^K C_k$. Using

$$\sum_{k=1}^K (C_k - \bar{C})^2 = \sum_{k=1}^K (C_k - \mu)^2 - K(\bar{C} - \mu)^2 \tag{15}$$

and $\mathrm{Var}(\bar{C}) = \mathrm{Var}(C)/K$, taking expectations and dividing by $K - 1$ proves Eq. 13. For Eq. 14, decompose

$$C - \mathbb{E}[C] = \big(C - \mathbb{E}[C \mid Z]\big) + \big(\mathbb{E}[C \mid Z] - \mathbb{E}[C]\big).$$

The two terms have zero expected product because the first has conditional mean zero given Z. Taking the expectation of the squared expression yields the result. □

The first term in Eq. 14 measures confidence variation among perturbations that produce the same trajectory. The second measures differences in mean confidence across decoded trajectories. This decomposition explains why RP can respond to behavioral changes that a fixed-trajectory analysis does not capture.

Two consequences clarify what this variance controls. First, for any $\tau > 0$, Chebyshev's inequality gives

$$\Pr(|C - \mathbb{E}[C]| \geq \tau) \leq \frac{\mathcal{S}_\sigma(V; X)}{\tau^2}. \tag{16}$$

Reducing the population variance therefore tightens a bound on confidence deviations from the perturbation-averaged value. Second, let Y_0 be the trajectory decoded under V, define $B = \mathbf{1}\{Z \neq Y_0\}$ and $p_{\mathrm{sw}} = \Pr(B = 1)$, and suppose $0 < p_{\mathrm{sw}} < 1$. With $\mu_{\mathrm{stay}} = \mathbb{E}[C \mid B = 0]$ and $\mu_{\mathrm{sw}} = \mathbb{E}[C \mid B = 1]$, the law of total variance gives

$$\mathrm{Var}_Z(\mathbb{E}[C \mid Z]) \geq p_{\mathrm{sw}}(1 - p_{\mathrm{sw}})(\mu_{\mathrm{sw}} - \mu_{\mathrm{stay}})^2. \tag{17}$$

Indeed, B is a function of Z, and the right-hand side is $\mathrm{Var}_B(\mathbb{E}[C \mid B])$. The variance thus captures confidence differences associated with trajectory switching; it does not measure switching probability alone.

We next examine the smooth sensitivity obtained when the trajectory is held fixed. This provides a local geometric interpretation of confidence variance without differentiating through decoding.

Proposition 2 (Local sensitivity on a fixed trajectory). Fix Y, let $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$, and define $g_Y = \nabla_V f_Y(V)$ and $H_Y = \nabla_V^2 f_Y(V)$. Suppose that, for $U \sim \mathcal{N}(0, I)$, the teacher-forced confidence admits the expansion

$$\begin{aligned} f_Y(V + \sigma U) = f_Y(V) &+ \sigma\langle g_Y, U\rangle_F \\ &+ \frac{\sigma^2}{2}\langle U, H_Y U\rangle_F + R_\sigma(U), \end{aligned} \tag{18}$$

where $\mathbb{E}[R_\sigma(U)^2] = \mathcal{O}(\sigma^6)$ as $\sigma \to 0$. Then

$$\mathrm{Var}_\epsilon[f_Y(V + \epsilon)] = \sigma^2 \|\nabla_V f_Y(V)\|_F^2 + \mathcal{O}(\sigma^4). \tag{19}$$

Proof. Write $A = \langle g_Y, U\rangle_F$ and $Q = \frac{1}{2}\langle U, H_Y U\rangle_F$. Gaussian symmetry gives

$$\mathbb{E}[A] = 0, \qquad \mathrm{Var}(A) = \|g_Y\|_F^2, \qquad \mathrm{Cov}(A, Q) = 0.$$

The last equality follows from the vanishing third-order centered Gaussian moments. In Eq. 18, the variance of $\sigma^2 Q$ is $\mathcal{O}(\sigma^4)$. By Cauchy–Schwarz, the covariance of σA with R_σ is also $\mathcal{O}(\sigma^4)$, while the remaining remainder terms are of higher order. Therefore,

$$\mathrm{Var}[f_Y(V + \sigma U)] = \sigma^2 \,\mathrm{Var}(A) + \mathcal{O}(\sigma^4),$$

which proves the claim. □

For sufficiently small perturbations, fixed-trajectory confidence variance is thus governed to leading order by the squared gradient norm. This result concerns the smooth function f_Y evaluated under unconditioned Gaussian perturbations. It is distinct from $\mathrm{Var}(C \mid Z = Y)$, where conditioning on the decoded trajectory changes the perturbation distribution. Proposition 1 describes the full re-decoded variance, while Proposition 2 explains local sensitivity when the trajectory is held fixed.

C.3 Sharpness-Aware Perturbation: Worst-Case Local Sensitivity

SAP evaluates the entropy loss along a gradient-guided perturbation while keeping the trajectory fixed. We analyze how this loss approximates the ideal worst-case objective within a radius ρ.

Proposition 3 (Approximation of the worst-case loss). Let $g = \nabla_V \ell(V)$ and define

$$\Delta_\rho(V) = \max_{\|\epsilon\|_F \le \rho} \big[\ell(V+\epsilon) - \ell(V)\big]. \tag{20}$$

Suppose that ℓ has a β-Lipschitz-continuous gradient on the perturbation ball. Then

$$|\Delta_\rho(V) - \rho\|g\|_F| \le \frac{\beta\rho^2}{2}. \tag{21}$$

For $g \neq 0$, the implemented perturbation $\epsilon^\star = \rho g/\|g\|_F$ satisfies

$$|\ell(V+\epsilon^\star) - \ell(V) - \rho\|g\|_F| \le \frac{\beta\rho^2}{2}. \tag{22}$$

Consequently, its loss differs from the ideal worst-case loss by at most

$$0 \le \max_{\|\epsilon\|_F \le \rho} \ell(V+\epsilon) - \ell(V+\epsilon^\star) \le \beta\rho^2. \tag{23}$$

Proof. For every ϵ in the ball, integrating the gradient along the segment from V to $V+\epsilon$ yields

$$\begin{aligned} &|\ell(V+\epsilon) - \ell(V) - \langle g, \epsilon\rangle_F| \\ &\quad \le \int_0^1 \|\nabla\ell(V+t\epsilon) - g\|_F \|\epsilon\|_F \, dt \le \frac{\beta}{2}\|\epsilon\|_F^2. \end{aligned} \tag{24}$$

Maximizing the resulting upper bound gives

$$\Delta_\rho(V) \le \rho\|g\|_F + \frac{\beta\rho^2}{2}.$$

When $g \neq 0$, evaluating the loss at $\epsilon^\star$ gives the lower bound

$$\Delta_\rho(V) \ge \ell(V+\epsilon^\star) - \ell(V) \ge \rho\|g\|_F - \frac{\beta\rho^2}{2}.$$

When $g = 0$, feasibility of $\epsilon = 0$ gives $0 \le \Delta_\rho(V) \le \beta\rho^2/2$. This proves Eq. 21. Substituting $\epsilon^\star$ into Eq. 24 proves Eq. 22. Combining the upper bound on the ideal loss with the lower bound on the implemented loss proves Eq. 23. □

Proposition 3 connects SAP directly to its implemented objective. For $g \neq 0$, the perturbed loss equals the original entropy loss plus $\rho\|g\|_F$ up to an error of order ρ^2. SAP therefore incorporates local sensitivity into confidence optimization, with a controlled approximation error under the stated smoothness assumption.

Effect of stop-gradient. The implementation treats $\epsilon^\star$ as constant during the outer update. Its gradient is therefore

$$\nabla_V^{\mathrm{sg}} \mathcal{L}_{\mathrm{SAP}} = \nabla_W \ell(W)|_{W=V+\epsilon^\star}. \tag{25}$$

Gradient Lipschitzness immediately gives

$$\|\nabla_V^{\mathrm{sg}} \mathcal{L}_{\mathrm{SAP}} - g\|_F \le \beta\rho. \tag{26}$$

Stop-gradient preserves the perturbed loss value and the bounds in Proposition 3 while avoiding derivatives through the construction of $\epsilon^\star$. The resulting update uses the gradient evaluated at the perturbed prefix.

C.4 Additional Analysis at Stationary Points

At stationary points, second-order expansions characterize the contribution of local curvature. We examine this geometry separately from the first-order SAP update.

Random perturbation. Under the assumptions of Proposition 2, if $\nabla_V f_Y(V) = 0$, then

$$\mathrm{Var}_\epsilon[f_Y(V+\epsilon)] = \frac{\sigma^4}{2}\|\nabla_V^2 f_Y(V)\|_F^2 + o(\sigma^4). \tag{27}$$

To see this, let $Q = \frac{1}{2}\langle U, H_Y U\rangle_F$ in Eq. 18. The linear term vanishes, and the remainder assumption gives

$$\mathrm{Var}[f_Y(V+\sigma U)] = \sigma^4 \mathrm{Var}(Q) + o(\sigma^4).$$

For a symmetric Hessian and a standard Gaussian vector, $\mathrm{Var}(U^\top H_Y U) = 2\|H_Y\|_F^2$. Hence $\mathrm{Var}(Q) = \frac{1}{2}\|H_Y\|_F^2$, proving Eq. 27.

Proposition 4 (Curvature of the ideal worst-case objective). Suppose that $\nabla_V \ell(V) = 0$ and that ℓ is twice continuously differentiable near V. Then, as $\rho \to 0$,

$$\Delta_\rho(V) = \frac{\rho^2}{2}\left[\lambda_{\max}\big(\nabla_V^2 \ell(V)\big)\right]_+ + o(\rho^2), \tag{28}$$

where $[a]_+ = \max(a, 0)$.

Proof. Let $H = \nabla_V^2 \ell(V)$. Since the gradient vanishes, Taylor expansion gives

$$\ell(V+\epsilon) - \ell(V) = \frac{1}{2}\langle \epsilon, H\epsilon\rangle_F + r(\epsilon),$$
$$\sup_{\|\epsilon\|_F \le \rho} |r(\epsilon)| = o(\rho^2).$$

Writing $\epsilon = \rho U$, we obtain

$$\Delta_\rho(V) = \frac{\rho^2}{2}\max_{\|U\|_F \le 1}\langle U, HU\rangle_F + o(\rho^2).$$

If $\lambda_{\max}(H) > 0$, the quadratic form is maximized by a unit eigenvector associated with $\lambda_{\max}(H)$. Otherwise, its maximum is zero, attained at $U = 0$. Thus the maximum equals $[\lambda_{\max}(H)]_+$, proving the result. □

Scope of the stationary-point result. Proposition 4 characterizes the ideal maximization over the perturbation ball. The implemented first-order SAP perturbation $\epsilon^\star = \rho g/\|g\|_F$ is analyzed in Proposition 3 for $g \neq 0$. At an exact stationary point, this expression is undefined and does not identify a leading Hessian eigenvector. The curvature result therefore describes the ideal robust objective, rather than the behavior of the implemented first-order update at stationarity.

C.5 Relationship Between the Two Local Objectives

The preceding analyses characterize local sensitivity from distributional and worst-case perspectives. To compare their geometric roles, consider a common smooth scalar function $h(V)$ with the trajectory held fixed, satisfying the respective assumptions above. Away from stationary points, as $\sigma \to 0$ and $\rho \to 0$, respectively,

$$\mathrm{Var}_\epsilon[h(V+\epsilon)] = \sigma^2\|\nabla_V h(V)\|_F^2 + \mathcal{O}(\sigma^4), \tag{29}$$

$$\max_{\|\epsilon\|_F \le \rho}[h(V+\epsilon) - h(V)] = \rho\|\nabla_V h(V)\|_F + \mathcal{O}(\rho^2), \tag{30}$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ in the first expression. Both quantities depend to leading order on the local gradient norm. Random perturbations aggregate variation across isotropic directions, whereas the worst-case objective measures the largest local increase.

At a stationary point, the corresponding second-order results relate these quantities to $\|H_h\|_F^2$ and $[\lambda_{\max}(H_h)]_+$, respectively, where $H_h = \nabla_V^2 h(V)$. These curvature measures satisfy

$$[\lambda_{\max}(H_h)]_+ \le \|H_h\|_2 \le \|H_h\|_F, \tag{31}$$

where $\|\cdot\|_2$ denotes the spectral norm. The variance depends on squared curvature across all directions, including negative curvature, while the ideal worst-case increase depends on the largest positive curvature.

Algorithm 1 TASCO with Random Perturbation

Require: Frozen LLM M_θ, test set $\mathcal{D}_{\text{test}}$, initial prefix V_0, optimizer $\mathcal{O}$, perturbation scale σ, number of perturbations $K \geq 2$, and weight λ_{rand}
Ensure: Optimized prefix V
Initialize $V \leftarrow V_0$
while not converged **do**
  Sample a mini-batch $\mathcal{B}$ from $\mathcal{D}_{\text{test}}$
  Generate trajectories $\mathbf{Y}_\mathcal{B}$ under V
  $\mathcal{L}_{\text{conf}} \leftarrow \mathcal{L}_\mathcal{B}(V; \mathbf{Y}_\mathcal{B})$
  for $k = 1, \ldots, K$ **do**
    Sample $\epsilon^{(k)} \sim \mathcal{N}(0, \sigma^2 I)$ and set $V^{(k)} \leftarrow V + \epsilon^{(k)}$
    Greedily generate trajectories $\mathbf{Y}_\mathcal{B}^{(k)}$ under $V^{(k)}$
    Compute $\{c_X^{(k)} : X \in \mathcal{B}\}$ using Eq. 3
  end for
  Compute $\mathcal{L}_{\text{rand}}$ from $\{c_X^{(k)}\}$ using Eq. 4
  $g \leftarrow \nabla_V \left(\mathcal{L}_{\text{conf}} + \lambda_{\text{rand}} \mathcal{L}_{\text{rand}}\right)$
  $V \leftarrow \mathcal{O}(V, g)$
end while
return V

The perturbation scales can also be related through

$$\sqrt{\mathbb{E}[\|\epsilon\|_F^2]} = \sigma\sqrt{Ld}. \tag{32}$$

Thus, $\sigma = \rho/\sqrt{Ld}$ matches the root-mean-square random perturbation norm to the SAP radius. This relation aligns their geometric scales without making the objectives equivalent.

In TASCO, RP operates on re-decoded trajectory-level log-probability, whereas SAP operates on the fixed-trajectory entropy loss. The common-function comparison therefore explains their different geometric interpretations without establishing numerical equivalence or an ordering between the implemented objectives. This result characterizes the ideal worst-case objective at stationarity; the implemented first-order perturbation is analyzed in Proposition 3 for nonzero gradients.

Summary. Together, these results characterize the local sensitivity targeted by TASCO's two stability objectives. RP penalizes confidence variation across nearby perturbed prefixes, including differences associated with changes in decoded trajectories. SAP approximates the worst-case fixed-trajectory entropy loss through a gradient-guided perturbation, with an approximation error bounded under the stated smoothness assumptions. These analyses explain how local stability enters confidence optimization; they do not guarantee answer correctness. The relationship between local stability and answer correctness is evaluated empirically in Appendix B.

D Complete Optimization Algorithms

This section provides the complete optimization procedures for the two TASCO variants. Both algorithms optimize a task-level prefix V over the unlabeled test set while keeping the LLM parameters θ frozen. They differ only in how local confidence stability is estimated and incorporated into prefix optimization.

Random Perturbation. Algorithm 1 presents the Random Perturbation variant. At each update, it samples K Gaussian perturbations around the current prefix and generates a trajectory under each perturbed prefix. The resulting trajectory-level confidence variance is averaged across the mini-batch and combined with the confidence loss to update V.

Sharpness-Aware Perturbation. Algorithm 2 presents the Sharpness-Aware Perturbation variant. It first generates one trajectory per input under the current prefix and holds these trajectories fixed throughout the update. A gradient-guided perturbation is then constructed from the confidence loss,

Algorithm 2 TASCO with Sharpness-Aware Perturbation

Require: Frozen LLM M_θ, test set $\mathcal{D}_{\text{test}}$, initial prefix V_0, optimizer $\mathcal{O}$, and perturbation radius ρ
Ensure: Optimized prefix V

```
Initialize V ← V_0
while not converged do
    Sample a mini-batch B from D_test
    Generate trajectories Y_B under V and hold them fixed
    g ← ∇_V L_B(V; Y_B)
    ε* ← ρ g / ||g||_F
    g_SAP ← ∇_V L_B(V + StopGrad(ε*); Y_B)
    V ← O(V, g_SAP)
end while
return V
```

Table 8: Accuracy (%) across MATH500 query difficulty groups.

	Qwen2.5-Math-7B		LLaMA-3.1-8B-Instruct	
Difficulty	Confidence Only	TASCO-RP	Confidence Only	TASCO-RP
Hard	46.7	54.4	9.7	14.4
Medium	75.4	80.8	48.9	54.3
Easy	94.9	96.0	90.1	88.5
Overall	71.0	75.8	47.8	50.2

after which the prefix is updated using the loss evaluated at the perturbed prefix. The perturbation is treated as constant during the second backward pass.

Random Perturbation obtains richer distributional information from multiple perturbed trajectories, whereas Sharpness-Aware Perturbation approximates worst-case local sensitivity with lower computational cost. In both cases, only the shared prefix is updated, and no labels, external verifiers, or auxiliary reward models are required.

E Further Analysis of Reasoning Effectiveness

E.1 Performance Across Query Difficulty

We estimate the difficulty of each MATH500 query using eight responses sampled from the unadapted backbone. Queries answered correctly in 0–2, 3–5, and 6–8 responses form the hard, medium, and easy groups, respectively. These groups contain 195, 130, and 175 queries for Qwen2.5-Math-7B, and 216, 92, and 192 queries for LLaMA-3.1-8B-Instruct. We evaluate Confidence Only and TASCO-RP on the same queries within each group.

Table 8 shows that TASCO-RP improves hard-query accuracy by 7.7 percentage points on Qwen2.5-Math-7B and 4.7 points on LLaMA-3.1-8B-Instruct. Medium-query accuracy increases by 5.4 points for both models. Changes on easy queries are smaller and mixed: +1.1 points on Qwen and -1.6 points on LLaMA. Thus, the overall gains are not driven solely by queries the unadapted models already answer reliably.

E.2 Generalization Across Mathematical Tasks

We evaluate prefix transfer on Qwen2.5-Math-7B across MATH500, AMC23, Minerva, and AIME24. For each source benchmark, we optimize a prefix and apply it to the other three benchmarks without further optimization. We compare its target-task accuracy with Zero-Shot CoT and a prefix optimized directly on the target benchmark.

Figure 8 shows that transferred prefixes outperform Zero-Shot CoT on all 12 source–target pairs. A prefix optimized on MATH500 raises AMC23 accuracy from 44.4% to 64.4%; one optimized on Minerva raises MATH500 accuracy from 52.2% to 71.2%. Although transfer usually falls short

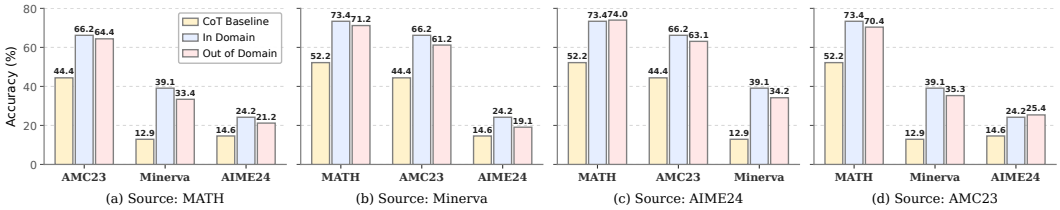


Figure 8: Cross-benchmark prefix transfer on Qwen2.5-Math-7B. CoT denotes Zero-Shot CoT; in-domain and out-of-domain denote prefixes optimized on the target and source benchmarks, respectively. Bars show accuracy (%).

of optimizing a prefix on the target benchmark, these gains persist without re-optimization. This suggests that TASCO learns task-level guidance that remains effective across different mathematical problem distributions. The prefix therefore supports more robust mathematical reasoning than would be expected from guidance tailored only to its source benchmark.

E.3 Semantic Evaluation of Reasoning Processes

Experimental Setup. We further evaluate how TASCO affects the intermediate reasoning process using DeepSeek-V4-Flash as an external judge. We compare paired trajectories generated by TASCO and confidence-only optimization using Qwen2.5-Math-7B on the full MATH500 and Minerva Math test sets, comprising 500 and 272 paired examples, respectively. For each example, the judge receives only the problem and the generated reasoning trajectory; no ground-truth answer is provided. Explicit final-answer spans are masked before evaluation to prevent the judge from inferring reasoning quality from answer correctness. We enable the model's thinking mode with high reasoning effort and request a structured JSON annotation for each trajectory.

The judge first divides each trajectory into semantically meaningful steps. Each step is labeled as correct, incorrect, or uncertain according to its mathematical validity, and as informative, redundant, or off-track according to whether it advances the solution. The judge also determines whether the trajectory commits prematurely to an answer or reasoning direction before sufficient support has been established. All aggregate metrics are computed locally from these annotations rather than directly scored by the judge.

Evaluation Metrics. For trajectory i with J_i semantic steps, let c_{ij} and u_{ij} denote the correctness and informativeness labels of step j, respectively. Let E_i denote the subset of steps labeled either correct or incorrect. We define Step Correctness as

$$\mathrm{SC}_i = \frac{\sum_{j \in E_i} \mathbf{1}\{c_{ij} = \text{correct}\}}{|E_i|}, \tag{33}$$

where $\mathbf{1}\{\cdot\}$ is the indicator function. Uncertain steps are excluded. A higher value indicates that a larger proportion of assessable reasoning steps are mathematically or logically valid.

Step Informativeness measures the proportion of steps that introduce useful information and advance the solution:

$$\mathrm{SI}_i = \frac{1}{J_i} \sum_{j=1}^{J_i} \mathbf{1}\{u_{ij} = \text{informative}\}\,. \tag{34}$$

Higher values indicate more productive reasoning.

Redundancy Rate measures the proportion of steps that mainly repeat or restate previously established information:

$$\mathrm{RR}_i = \frac{1}{J_i} \sum_{j=1}^{J_i} \mathbf{1}\{u_{ij} = \text{redundant}\}\,. \tag{35}$$

Lower values indicate less repetitive reasoning.

Premature Commitment is a trajectory-level indicator of whether the model commits to an answer or solution direction before sufficient justification and subsequently treats it as fixed. Premature

Table 9: Reasoning-process evaluation using DeepSeek-V4-Flash as the external judge. All values are percentages.

Model	Method	MATH500				Minerva Math			
		SC ↑	SI ↑	PC ↓	RR ↓	SC ↑	SI ↑	PC ↓	RR ↓
Qwen2.5-Math-7B	Confidence Only	82.7	77.6	18.8	15.1	75.4	75.7	22.8	11.2
	TASCO-SAP	85.2	80.4	17.2	11.6	77.5	80.5	19.5	10.1

Commitment Rate is the proportion of trajectories exhibiting this behavior. Lower values indicate that the model is less likely to narrow its reasoning direction prematurely.

Results. As shown in Table 9, TASCO consistently improves all four reasoning-process metrics across both datasets. It produces a larger proportion of correct and informative steps while reducing redundant reasoning and premature commitment. These results suggest that stability-aware optimization affects not only final-answer accuracy but also how the model develops and maintains its reasoning process.

Table 10: Qualitative comparison on polynomial root finding. The confidence-only trajectory divides by a potentially zero factor and consequently discards the valid root $x = 5$. Python verification code is omitted for brevity; the remaining reasoning follows the original model outputs.

QUESTION: Find the roots of $(x-3)^3 + (x-7)^3 = (2x-10)^3$.

GROUND TRUTH: 3,5,7

CONFIDENCE-ONLY REASONING (INCORRECT):

To solve the equation

$$(x-3)^3 + (x-7)^3 = (2x-10)^3,$$

we can use the identity for the sum of cubes:

$$a^3 + b^3 = (a+b)(a^2 - ab + b^2).$$

Let $a = x-3$ and $b = x-7$. Using the identity, we get

$$(2x-10)\left((x-3)^2 - (x-3)(x-7) + (x-7)^2\right) = (2x-10)^3.$$

Dividing both sides by $2x-10$, we get

$$(x-3)^2 - (x-3)(x-7) + (x-7)^2 = (2x-10)^2.$$

Expanding both sides, we get

$$x^2 - 6x + 9 - (x^2 - 10x + 21) + x^2 - 14x + 49 = 4x^2 - 40x + 100.$$

Simplifying the left-hand side gives

$$x^2 - 10x + 37 = 4x^2 - 40x + 100.$$

Rearranging the terms, we get

$$3x^2 - 30x + 63 = 0.$$

Dividing both sides by 3, we obtain

$$x^2 - 10x + 21 = 0.$$

Factoring the quadratic equation gives

$$(x-3)(x-7) = 0.$$

Therefore, the roots of the equation are $x = 3$ and $x = 7$.
The roots of the equation are stated to be $x = 3$, $x = 7$, and $x = 8 - \sqrt{21}i$. Since we are looking for real roots, the real roots are $x = 3$ and $x = 7$.
Therefore, the final answer is

$$\boxed{3, 7}.$$

TASCO REASONING (CORRECT):

To solve the equation

$$(x-3)^3 + (x-7)^3 = (2x-10)^3,$$

First, let $y = x - 5$. Then the equation becomes

$$(y+2)^3 + (y-2)^3 = (2y)^3.$$

Expanding both sides, we get

$$(y^3 + 6y^2 + 12y + 8) + (y^3 - 6y^2 + 12y - 8) = 8y^3.$$

Combining like terms gives

$$2y^3 + 24y = 8y^3.$$

Rearranging terms, we obtain

$$6y^3 - 24y = 0.$$

Factoring out $6y$ gives

$$6y(y^2 - 4) = 0.$$

This gives three solutions:

$$y = 0, \qquad y = 2, \qquad y = -2.$$

Substituting back $y = x - 5$, we get

$$x - 5 = 0 \implies x = 5, x - 5 = 2 \implies x = 7, and x - 5 = -2 \implies x = 3.$$

Thus, the roots of the equation are $x = 3, 5, 7$.
The roots of the equation are

$$\boxed{3, 5, 7}.$$

Table 11: A representative TASCO reasoning example on AIME24. The textual `qwen25-math-cot` prompt is shown in role-structured form; the learned continuous prefix operates at the embedding level. Generated solver code is omitted for brevity.

PROMPT INPUT (QWEN25-MATH-COT)

System: Please reason step by step, and put your final answer within \boxed{}.

User: Every morning Aya goes for a 9-kilometer-long walk and stops at a coffee shop afterwards. When she walks at a constant speed of s kilometers per hour, the walk takes her 4 hours, including t minutes spent in the coffee shop. When she walks at $s+2$ kilometers per hour, the walk takes her 2 hours and 24 minutes, including t minutes spent in the coffee shop. Suppose Aya walks at $s+\frac{1}{2}$ kilometers per hour. Find the number of minutes the walk takes her, including the t minutes spent in the coffee shop.

GROUND TRUTH

$\boxed{204}$

TASCO REASONING OUTPUT

To solve this problem, we first determine the values of s and t using the two given walking times.

Step 1: Formulate the first time constraint. When Aya walks at s kilometers per hour, the walking time is $\frac{9}{s}$ hours, while the time spent in the coffee shop is $\frac{t}{60}$ hours. Since the total time is 4 hours,

$$\frac{9}{s}+\frac{t}{60}=4.$$

Step 2: Formulate the second time constraint. When Aya walks at $s+2$ kilometers per hour, the total duration is 2 hours and 24 minutes, or 2.4 hours. Therefore,

$$\frac{9}{s+2}+\frac{t}{60}=2.4.$$

Step 3: Solve for s and t. Solving the two equations gives

$$(s,t)=(-3,240) \qquad \text{or} \qquad (s,t)=(2.5,24).$$

Because a speed of -3 kilometers per hour is not physically meaningful, we discard the first solution. Hence,

$$s=2.5, \qquad t=24.$$

Step 4: Compute the requested duration. At the new speed,

$$s+\frac{1}{2}=2.5+0.5=3$$

kilometers per hour. Walking 9 kilometers therefore takes

$$\frac{9}{3}=3$$

hours. Including the 24 minutes spent in the coffee shop, the total duration is

$$3\times 60+24=204$$

minutes.
Therefore, the walk takes $\boxed{204}$ minutes.