IS SHARPER ALWAYS BETTER? RETHINKING THE ROLE OF SHARPNESS IN POWER SAMPLING FOR LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Power Sampling (PS) is a training-free inference-time method that improves language model reasoning by sharpening the trajectory-level distribution. While recent studies have developed effective methods for sampling from the power distribution with a fixed sharpening strength α, the impact of α remains largely unexplored. In this work, we first systematically investigate the role of the sharpening strength α in PS across diverse models and reasoning tasks, demonstrating that reasoning performance varies non-monotonically with α across models and tasks. We further analyze the effect of α through a preservation–recovery decomposition, which shows that larger α improves the preservation of correct solutions favored by the base model, but reduces the recovery of alternative correct trajectories when these dominant trajectories are misleading. These opposing trends provide empirical evidence for a concentration–exploration trade-off governed by α. Based on these observations, we propose Adaptive Concentration–Exploration Power Sampling (ACE), a training-free decoding framework built upon the Sequential Monte Carlo method that adaptively controls α for each problem instance according to its evolving particle population during inference. By adapting the sharpening strength throughout generation, ACE dynamically balances concentration on promising trajectories with the preservation of diverse reasoning paths. Experiments across diverse reasoning benchmarks demonstrate that ACE consistently improves reasoning performance through adaptive sharpening without additional training or external reward signals.

1 INTRODUCTION

Large language models (LLMs) have demonstrated remarkable capabilities in solving a wide range of complex tasks, including mathematical reasoning, code generation, and multi-step decision-making (Hurst et al., 2024; Guo et al., 2025; Yang et al., 2025). Beyond scaling pre-training, recent studies suggest that further improvements in reasoning can also arise from effectively sharpening the generation distribution of existing models (Yue et al., 2025; Shao et al., 2025; Fan et al., 2026; Karan & Du, 2026). Both post-training approaches, such as reinforcement learning, and inference-time strategies can improve reasoning by shifting probability mass toward more desirable reasoning trajectories while suppressing less effective ones. Among these inference-time strategies, Power Sampling (PS) provides a training-free framework that explores distribution sharpening as a mechanism for improving reasoning performance (Karan & Du, 2026).

Rather than updating model parameters or evaluating trajectories with an external reward, PS constructs a sharpened trajectory-level distribution solely based on the base model's own probability distribution: $\pi_\alpha(y|x) \propto p_\theta(y|x)^\alpha$, where $p_\theta(y|x)$ is the base model distribution and $\alpha \geq 1$ controls the strength of sharpening. Larger α concentrates more probability mass on high-probability trajectories, whereas smaller α preserves a flatter distribution with greater exploration. As the base model may already assign substantial probability mass to high-quality reasoning trajectories, PS further concentrates probability on these trajectories, thereby shifting generation toward high-quality regions. To sample from the power distribution, recent works have adopted a range of sampling techniques, including the Metropolis-Hastings algorithm (Metropolis et al., 1953; Karan & Du, 2026), the Sequential Monte Carlo method (Del Moral et al., 2006; Azizi et al., 2026) and the Twisted

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.

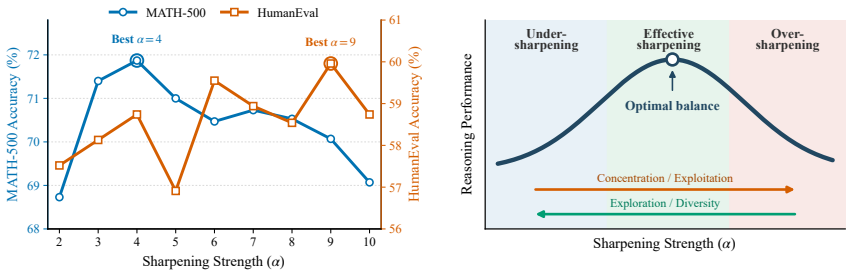


Figure 1: Left: Reasoning performance of Qwen2.5-7B exhibits a non-monotonic relationship with sharpening strength α, with different tasks favoring different levels of sharpening. Right: Illustration of the concentration–exploration trade-off induced by α in PS.

Monte Carlo method (Ji et al., 2026). The empirical success of these methods demonstrates that PS can substantially improve reasoning by sharpening the generation distribution, without additional training or external reward signals.

The sharpening strength α plays a crucial role in determining how concentrated the trajectory-level distribution becomes. By amplifying relative probability differences among trajectories, α controls the balance between concentrating probability mass on high-probability paths and preserving alternative reasoning trajectories. Despite its importance, the role of α remains largely underexplored. Existing studies have only conducted limited ablation analyses on α (Karan & Du, 2026), and most PS approaches adopt a single fixed exponent (e.g., $\alpha = 4$) across different models and reasoning tasks (Ji et al., 2026; Azizi et al., 2026; Guo et al., 2026). However, since different models and tasks may exhibit distinct probability landscapes, the effective choice of α may also vary across reasoning scenarios. For example, when a model's high-probability trajectories are more likely to correspond to correct reasoning solutions, increasing α may improve sampling efficiency by concentrating probability mass on reliable trajectories. In contrast, when high-probability trajectories are less reliable or alternative solutions are necessary, a smaller α may better preserve exploration over diverse reasoning trajectories (Chen et al., 2026b).

Motivated by this intuition, we conduct a systematic diagnostic study on the role of sharpening strength α in PS (Section 4). Empirically, we find that reasoning performance exhibits a non-monotonic relationship with α, and that the optimal exponent varies across models and tasks (Figure 1 (Left)). Through a preservation–recovery decomposition of trajectory quality, we further characterize how the sharpening strength α shapes a **concentration–exploration trade-off** (Figure 1 (Right)). While larger α improves the exploitation of dominant high-probability trajectories, it reduces the ability to recover correct alternatives when these trajectories correspond to misleading reasoning paths. Thus, α is not merely a sharpening hyperparameter, but a control variable that governs the balance between concentration and exploration in PS. Since the appropriate sharpening strength is not known in advance and may change as generation unfolds, we seek to regulate this balance during inference from a given initial α.

Motivated by these findings, we propose **A**daptive **C**oncentration–**E**xploration Power Sampling (**ACE**), a training-free trajectory-level decoding framework that adaptively controls the sharpening strength of PS during generation. Starting from a given initial exponent, ACE adjusts α for each problem instance according to the evolving particle population, using effective sample size to constrain weight concentration and token-level entropy to trigger additional updates. This mechanism allows the sharpening strength to be revised throughout inference to regulate the concentration–exploration trade-off. Our main contributions are summarized as follows:

- We systematically investigate the effect of the sharpening strength α in PS across diverse models and reasoning tasks. We further introduce a preservation–recovery decomposition to characterize its behavior, providing empirical evidence for a concentration–exploration trade-off associated with α: stronger sharpening better preserves dominant trajectories but reduces the ability to recover alternative solutions when those trajectories are misleading.
- Based on these observations, we propose **Adaptive Concentration–Exploration Power Sampling (ACE)**, a training-free inference framework that adaptively controls α for each problem instance according to the evolving particle population during generation.

- Experiments across multiple reasoning benchmarks and models show that ACE achieves strong overall performance and improves over the fixed-α PS baselines in most settings.

2 Related Work

Test-Time Scaling for LLM Reasoning. Test-time scaling improves LLM reasoning by increasing inference-time computation without updating model parameters. Representative approaches improve reasoning by generating multiple candidate solutions and selecting or aggregating them, including self-consistency (Wang et al., 2023), Best-of-N sampling (Cobbe et al., 2021), verifier-based selection (Lightman et al., 2023) and tree-based methods (Yao et al., 2023). Recently, Power Sampling has emerged as a new training-free approach that directly sharpens the trajectory-level model distribution, targeting $\pi_\alpha(y|x) \propto p_\theta(y|x)^\alpha$ (Karan & Du, 2026). Subsequent work develops more efficient approximations to this distribution through autoregressive sampling and Sequential Monte Carlo (Ji et al., 2026; Azizi et al., 2026). Different from these methods, which mainly study how to sample efficiently from a specified power distribution, ACE focuses on how the sharpening strength α itself should be adaptively controlled during generation. A more comprehensive discussion of related work is included in Appendix B.

3 Trajectory-Level Distribution Sharpening via Power Sampling

Let x denote the prompt and $y = (y_1, \ldots, y_T)$ denote the token trajectory. An autoregressive language model parameterized by θ defines a trajectory-level probability distribution by factorizing the probability of each token conditioned on its preceding tokens: $p_\theta(y|x) = \prod_{t=1}^{T} p_\theta(y_t|y_{<t}, x)$, where $y_{<t}$ denotes the token prefix before position t.

PS constructs a sharpened distribution by applying a power transformation over the trajectory: $\pi_\alpha(y|x) = \frac{p_\theta(y|x)^\alpha}{Z_\alpha(x)}$, where $Z_\alpha(x) = \sum_v p_\theta(v|x)^\alpha$ is the normalization constant and α controls the sharpening strength. When $\alpha = 1$, the transformed distribution is identical to the original model distribution. For $\alpha > 1$, the transformation sharpens the trajectory-level distribution by amplifying probability differences among trajectories, assigning relatively more probability mass to trajectories with high probability under the base model and less probability mass to low-probability trajectories.

PS differs from low-temperature sampling in the level of distribution being sharpened. Specifically, in low-temperature sampling, the next-token distribution is obtained by locally sharpening the conditional distribution at each decoding step: $p_\tau^{\text{temp}}(y_t|y_{<t}, x) = \frac{p_\theta(y_t|y_{<t},x)^{1/\tau}}{\sum_{y_t} p_\theta(y_t|y_{<t},x)^{1/\tau}}$, where τ is the token-level temperature. The probability of the next token depends only on the current prefix and the local token probabilities. In contrast, PS defines the trajectory-level distribution $\pi_\alpha(y|x) \propto p_\theta(y|x)^\alpha$ and the induced next-token distribution is obtained by marginalizing over all possible future continuations: $\pi_\alpha(y_t|y_{<t}, x) = \frac{\sum_{y_{>t}} p_\theta(y_{<t},y_t,y_{>t}|x)^\alpha}{\sum_{y_{\geq t}} p_\theta(y_{<t},y_{\geq t}|x)^\alpha}$.

Although both approaches increase distribution concentration, they operate on fundamentally different objects. Low-temperature sampling reinforces local token preferences conditioned on the current prefix, whereas PS incorporates information from complete trajectories and reshapes the distribution over global reasoning paths.

The sharpening strength α therefore plays a critical role in determining the effect of trajectory-level sharpening. While larger α increasingly emphasizes high-probability trajectories under the base model, excessive sharpening may reduce the diversity of alternative reasoning paths.

4 The Concentration–Exploration Trade-off in Power Sampling

In this section, we investigate how the sharpening strength α in PS affects reasoning performance across models and tasks. We first evaluate PS with different α values to characterize how reasoning performance changes with the degree of sharpening, and then introduce a preservation–recovery decomposition to characterize an empirical concentration–exploration trade-off associated with the sharpening strength.

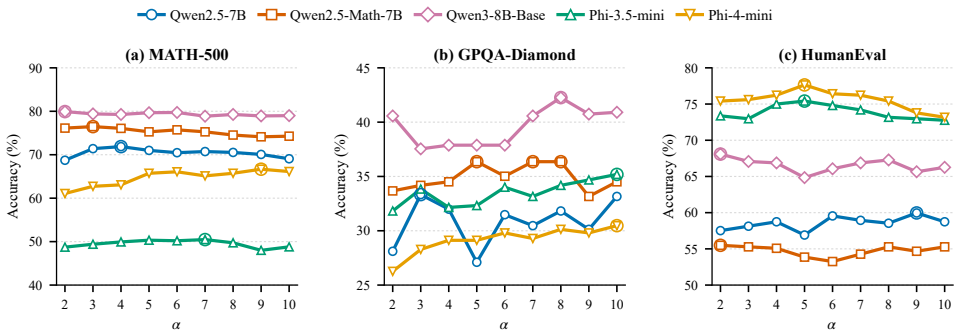


Figure 2: Performance varies non-monotonically with α across five models and three benchmarks.

4.1 Stronger Sharpening Does Not Always Improve Performance

We systematically evaluate the effect of the sharpening strength α in PS by varying α from 2 to 10. We evaluate it on three benchmarks, including MATH-500 (Hendrycks et al., 2021), GPQA-Diamond (Rein et al., 2023), and HumanEval (Chen et al., 2021), using five models: Qwen2.5-7B (Yang et al., 2024a), Qwen2.5-Math-7B (Yang et al., 2024b), Qwen3-8B-Base (Yang et al., 2025), Phi-3.5-mini (Abdin et al., 2024a), and Phi-4-mini (Abdin et al., 2024b).

As shown in Figure 2, the effect of sharpening strength α is not monotonic across models and tasks. Increasing α can improve reasoning performance in some settings, but stronger sharpening does not consistently lead to further gains.

Across different benchmarks, the performance curves exhibit diverse behaviors. On MATH-500, most models remain relatively stable across a broad range of α, with moderate sharpening often achieving competitive performance. In contrast, GPQA-Diamond shows larger fluctuations as α varies, suggesting that the effect of sharpening depends strongly on the reasoning landscape of each task. HumanEval also exhibits model-specific trends, where some models benefit from stronger sharpening while others reach peak performance at smaller values of α. These results demonstrate that the sharpening strength cannot be determined by a universal rule across models and tasks.

4.2 A Preservation–Recovery Decomposition of Sharpening Effects

To better understand the mechanism behind the non-monotonic performance observed in Figure 2, we introduce a preservation–recovery decomposition based on the outcomes under a strong sharpening strength. Let $C_\alpha(x) \in \{0, 1\}$ indicate whether PS with exponent α correctly solves problem x. We use $\alpha = 10$ as a diagnostic reference, representing a strong sharpening setting. For each model and each benchmark, we partition the whole dataset into:

$$\mathcal{D}_{10}^{+} = \{x : C_{10}(x) = 1\}, \quad \mathcal{D}_{10}^{-} = \{x : C_{10}(x) = 0\}. \tag{1}$$

The first group contains instances correctly solved under strong sharpening, while the second group contains instances where strong sharpening fails to produce a correct solution.

We then evaluate the effect of different sharpening strengths on these two fixed subsets:

$$\text{Preservation}(\alpha) = \frac{1}{|\mathcal{D}_{10}^{+}|} \sum_{x \in \mathcal{D}_{10}^{+}} C_\alpha(x), \quad \text{Recovery}(\alpha) = \frac{1}{|\mathcal{D}_{10}^{-}|} \sum_{x \in \mathcal{D}_{10}^{-}} C_\alpha(x). \tag{2}$$

Preservation measures the fraction of instances solved by strong sharpening that remain correctly solved under the current α, whereas recovery measures the ability to recover correct solutions missed by strong sharpening by exploring trajectories beyond the concentrated region.

Figure 3 reveals two opposing effects associated with different α. Across three different tasks, preservation generally improves as sharpening increases, indicating that larger α increasingly preserves solutions obtained under strong sharpening. In contrast, recovery exhibits the opposite trend: smaller α retain a greater ability to recover correct solutions missed by strong sharpening, whereas stronger sharpening increasingly reduces access to such alternatives. These trends are consistently observed across different model families and task domains.

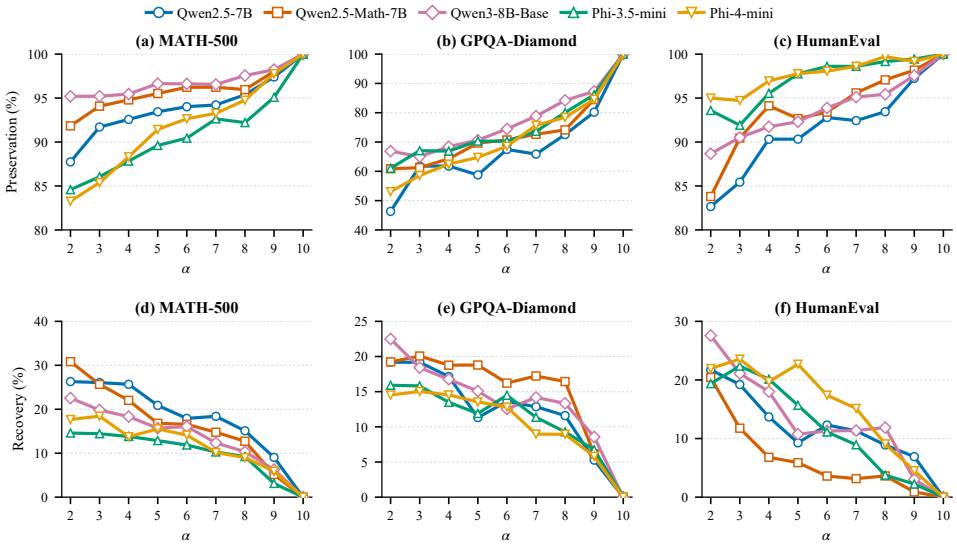


Figure 3: Preservation and recovery under the partition defined by the outcomes of $\alpha = 10$. The top row reports preservation on instances correctly solved by strong sharpening, while the bottom row reports recovery on instances not solved by strong sharpening.

The decomposition provides an explanation for the non-monotonic performance observed in Figure 2. Let $\rho = |\mathcal{D}_{10}^{+}|/|\mathcal{D}|$ denote the fraction of instances correctly solved under strong sharpening. The overall performance can be decomposed as

$$\mathrm{Perf}(\alpha) = \rho\,\mathrm{Preservation}(\alpha) + (1-\rho)\,\mathrm{Recovery}(\alpha), \tag{3}$$

which reflects a balance between preservation and recovery. As α increases, the model concentrates on high-probability trajectories and reduces the exploration of alternative viable trajectories, leading to the non-monotonic performance trend.

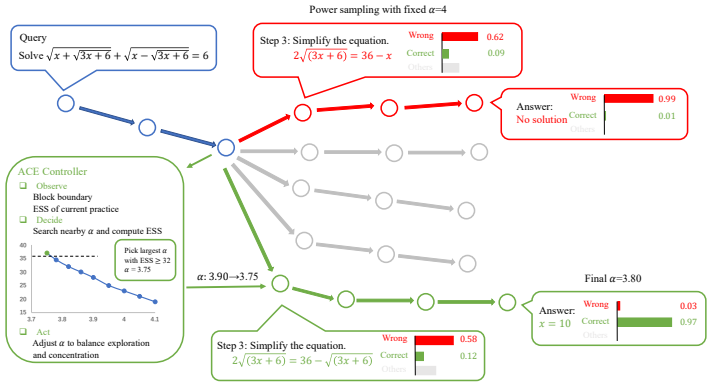


Figure 4: Illustrative comparison between fixed-α power sampling and ACE.

Since trajectory correctness is unavailable during generation and the appropriate sharpening strength cannot be determined beforehand, practical implementations typically rely on a prespecified α. Starting from a given initial α, we therefore seek to adapt the sharpening strength to the evolving state of generation. This adaptation balances concentration on high-probability trajectories with exploration of alternative reasoning paths, analogous to the exploitation–exploration trade-off in reinforcement learning. Increasing α reinforces concentration on currently likely trajectories, while decreasing it allows alternative paths to remain accessible. Figure 4 illustrates this balance at the instance level: a fixed α over-concentrates on an initially dominant incorrect trajectory, whereas adaptive sharpening preserves an alternative path that eventually leads to the correct answer.

5 ADAPTIVE CONCENTRATION–EXPLORATION POWER SAMPLING

In this section, we introduce Adaptive Concentration–Exploration Power Sampling (ACE), a training-free framework that adaptively controls the sharpening strength for each problem instance during Sequential Monte Carlo inference. Starting from a given initial α, ACE allows different instances to follow different α_t according to the evolving state of their particle populations.

5.1 PRELIMINARIES: SEQUENTIAL MONTE CARLO FOR POWER SAMPLING

In PS, Sequential Monte Carlo (SMC) (Del Moral et al., 2006; Azizi et al., 2026) approximates the trajectory-level power distribution $\pi_\alpha(y|x) \propto p_\theta(y|x)^\alpha$ with a population of weighted particles $\{y_{\le t}^{(i)}, w_t^{(i)}\}_{i=1}^N$, where $w_0^{(i)} = 1/N$. Particles are generated autoregressively from a proposal distribution q_t, e.g., the base-model distribution $p_\theta(y_t|y_{<t}, x)$, and the intermediate target is $\gamma_t(y_{\le t}|x) \propto p_\theta(y_{\le t}|x)^\alpha$. During generation step t, weights of these particles are updated by

$$w_t^{(i)} = w_{t-1}^{(i)} \frac{\gamma_t(y_{\le t}^{(i)}|x)}{\gamma_{t-1}(y_{<t}^{(i)}|x) q_t(y_t^{(i)}|y_{<t}^{(i)}, x)}. \tag{4}$$

Weight degeneracy is measured by the effective sample size: $\mathrm{ESS}_t = (\sum_{i=1}^N w_t^{(i)})^2 / \sum_{i=1}^N (w_t^{(i)})^2$. ESS reflects the concentration of the particle weights: a more balanced distribution across particles yields a larger ESS, whereas greater concentration on a small number of trajectories yields a smaller ESS. To mitigate particle degeneracy, SMC performs resampling when the ESS falls below a predefined threshold and resets the weights to $1/N$.

5.2 DYNAMIC ALPHA ADAPTATION

ACE replaces the fixed exponent α with a time-varying exponent α_t, defining the intermediate target as $\gamma_t^{\alpha_t}(y_{\le t}|x) \propto p_\theta(y_{\le t}|x)^{\alpha_t}$, where α_t controls the concentration of the target distribution. The following proposition formalizes the relationship between ESS and α_t.

Proposition 1. *For a fixed particle set $\{y_{\le t}^{(i)}\}_{i=1}^N$ with positive trajectory probabilities, if the proposal distribution is the base-model distribution and no resampling has occurred, then the effective sample size is non-increasing with respect to the sharpening strength α for $\alpha \ge 1$. Specifically, for any $1 \le \alpha_1 < \alpha_2$,*

$$\mathrm{ESS}_t(\alpha_1) \ge \mathrm{ESS}_t(\alpha_2).$$

The equality holds when all particles have identical trajectory probabilities under $p_\theta(y_{1:t}|x)$.

Proposition 1 motivates using ESS as a criterion for adapting the sharpening strength.

Adaptive α Selection. Given the current exponent α_t, ACE constructs a local candidate set:

$$\mathcal{A}_t = \{\alpha \mid \alpha \in [\max(1, \alpha_t - \Delta), \alpha_t + \Delta]\},$$

where Δ controls the adaptation range. For each candidate exponent $\alpha_c \in \mathcal{A}_t$, ACE estimates the corresponding importance weights under the updated target distribution:

$$\log w_t^{(i)}(\alpha_c) = \log w_t^{(i)} + (\alpha_c - \alpha_t) \log p_\theta(y_{\le t}^{(i)}|x). \tag{5}$$

The corresponding ESS is computed as: $\mathrm{ESS}_t(\alpha_c) = \frac{\left(\sum_{i=1}^N w_t^{(i)}(\alpha_c)\right)^2}{\sum_{i=1}^N (w_t^{(i)}(\alpha_c))^2}$. ACE selects the largest candidate satisfying the ESS constraint:

$$\alpha_{t+1} = \max_{\alpha_c \in \mathcal{A}_t} \alpha_c \quad \text{s.t.} \quad \mathrm{ESS}_t(\alpha_c) \ge \tau_{\mathrm{ESS}} N, \tag{6}$$

where $\tau_{\mathrm{ESS}} \in (0, 1)$ is the minimum required ESS ratio.

For each instance, this adaptive rule encourages stronger concentration whenever its current particle population contains sufficient diversity, while automatically reducing the sharpening strength when the particle distribution becomes overly concentrated. Therefore, ACE dynamically regulates the concentration–exploration trade-off by adapting the sharpening strength.

Algorithm 1 Adaptive Concentration–Exploration Power Sampling (ACE)

Require: Prompt x, particles N, initial exponent α_0, generation limit T.
Initialize particles, uniform weights, and uncertainty states.
for $t = 1, \ldots, T$ **do**
    Sample from q_t and update weights using Equation (4).
    Update particle uncertainty states using token-level entropy.
    if a block boundary or uncertainty trigger is reached **then**
        Update α according to Equation (6).
        if ESS is smaller than the resampling threshold **then**
            Resample and reset weights to uniform.
        end if
    end if
end for
return a completion sampled from the terminal weighted particles.

When to Update α. Following previous studies (Karan & Du, 2026; Azizi et al., 2026; Ji et al., 2026), ACE adopts a block-wise update strategy to balance adaptation frequency and computational cost. Specifically, the generation process is partitioned into blocks of B tokens and ACE updates α at the end of each block. At each block boundary, ACE evaluates the current particle population, selects the new exponent according to the ESS-constrained adaptation rule, and performs resampling if the ESS falls below the resampling threshold. This periodic update schedule provides stable adaptation while avoiding the computational overhead of updating α at every decoding step.

In addition to periodic updates, ACE leverages token-level uncertainty to identify potentially important adaptation points within a block. Recent studies have shown that high-entropy tokens often correspond to decision points where the model exhibits uncertainty among multiple plausible continuations, which may lead to substantial divergence among subsequent trajectories (Li et al., 2025; Wang et al., 2026). Motivated by this phenomenon, ACE introduces an uncertainty-aware mechanism for determining when to update α based on the state of the particle population. At each generation step, ACE monitors the token-level entropy of every particle. When a particle encounters a high-entropy token, it is marked as an uncertainty candidate and becomes mature after generating an additional L tokens. Once the number of mature candidates reaches $\tau_{\mathrm{mat}} N$, ACE performs an additional α update using the same ESS-constrained rule. This particle-level uncertainty accumulation mechanism allows ACE to capture important regions where future trajectories may diverge.

Overall, ACE dynamically adjusts the sharpening strength according to the evolving particle population during SMC inference. By combining ESS-constrained adaptation with block-wise scheduling and particle-level uncertainty-aware triggering, ACE provides a flexible mechanism to regulate the concentration–exploration trade-off of PS. The overall algorithm is summarized in Algorithm 1.

6 Experiments

6.1 Setup

Benchmarks. We evaluate ACE on three reasoning and code generation benchmarks: HumanEval (Chen et al., 2021), GPQA-Diamond (Rein et al., 2023), and MATH-500 (Hendrycks et al., 2021). HumanEval evaluates code generation, while GPQA-Diamond and MATH-500 evaluate scientific and mathematical reasoning, respectively.

Models. We evaluate ACE on five language models: Qwen2.5-7B (Yang et al., 2024a), Qwen2.5-Math-7B (Yang et al., 2024b), Qwen3-8B-Base (Yang et al., 2025), Phi-3.5-mini (Abdin et al., 2024a) and Phi-4-mini (Abdin et al., 2024b).

Baselines. We compare ACE with standard decoding, low-temperature sampling and recent PS methods. **Standard** decoding directly samples from the original model distribution without additional inference-time modification, while low-temperature (**low-temp**) sampling applies temperature scaling to the next-token distribution. We consider the following PS methods that construct samples from the power distribution using different strategies: the Metropolis–Hastings (**MH**) method (Metropolis et al., 1953; Karan & Du, 2026), the Sequential Monte Carlo (**SMC**) method (Del Moral et al., 2006; Azizi et al., 2026), and the Twisted Monte Carlo (**TMC**) method (Ji et al., 2026).

Table 1: Accuracy (%) across HumanEval, GPQA-Diamond, and MATH-500. Bold values indicate the highest accuracy for each model and benchmark.

Model	Method	HumanEval	GPQA-Diamond	MATH-500	Avg.
Qwen2.5-7B	Standard	29.3	28.3	39.7	32.4
	Low-temp	56.0	29.7	63.7	49.8
	MH	57.2	31.4	69.5	52.7
	TMC	54.2	32.0	62.9	49.7
	SMC	58.1	32.1	**72.5**	54.2
	ACE	**62.3**	**33.1**	71.2	**55.5**
Qwen2.5-Math-7B	Standard	28.2	30.5	48.4	35.7
	Low-temp	51.8	35.2	66.3	51.1
	MH	55.4	35.7	71.4	54.2
	TMC	52.0	**36.1**	68.0	52.0
	SMC	55.0	33.4	76.1	54.9
	ACE	**57.4**	33.6	**77.1**	**56.1**
Qwen3-8B-Base	Standard	42.6	34.3	59.4	45.4
	Low-temp	65.6	40.4	77.7	61.2
	MH	67.3	40.7	**79.8**	62.6
	TMC	67.1	39.3	78.0	61.5
	SMC	67.7	37.4	79.4	61.5
	ACE	**70.0**	**41.0**	79.5	**63.5**
Phi-3.5-mini	Standard	58.7	33.1	42.0	44.6
	Low-temp	70.2	33.5	45.8	49.8
	MH	71.2	33.9	48.6	51.3
	TMC	69.4	**35.2**	45.1	49.9
	SMC	**74.4**	32.3	49.3	52.0
	ACE	74.3	33.4	**49.7**	**52.5**
Phi-4-mini	Standard	50.9	30.6	51.8	44.4
	Low-temp	70.5	32.9	66.0	56.5
	MH	74.0	**34.2**	**68.4**	**58.9**
	TMC	71.8	34.0	67.0	57.6
	SMC	76.7	29.2	63.0	56.3
	ACE	**77.6**	30.6	66.2	58.1

Implementation Details. All methods are evaluated in a single-shot setting, with performance averaged over five independent runs. We set the temperature of the low-temperature baseline to 0.25. Following prior work (Karan & Du, 2026; Azizi et al., 2026; Ji et al., 2026), we fix $\alpha = 4$ for the PS baselines and initialize ACE at the same value to assess adaptation during generation relative to this conventional fixed α setting. Moreover, all PS methods use a tempered base-model distribution as the proposal, with temperature $1/\alpha$ for the fixed-α baselines and $1/\alpha_t$ for ACE. All PS methods use a block size of $B = 192$ tokens, and all methods share a maximum generation length of 3072 tokens across tasks. Additional implementation details are provided in Appendix C.

6.2 Results

As shown in Table 1, ACE delivers strong and consistent performance across models and benchmarks, achieving the highest average accuracy on four of the five models. Compared with the evaluated PS methods that use a fixed $\alpha = 4$, ACE performs better in most settings. In particular, ACE improves upon SMC in most evaluated settings, supporting the benefit of adapting the sharpening strength relative to this fixed-exponent configuration. Overall, ACE exhibits more robust performance across different models and tasks. Additional results are reported in Appendix D.

As shown in Figure 5 (left), ACE adapts the sharpening strength differently across models and tasks. Distributions of final α_T vary in both central tendency and dispersion: GPQA-Diamond generally shifts toward smaller α, MATH-500 is concentrated around the initialization α_0, whereas HumanEval tends toward larger α with a substantially wider range. The variation within each setting suggests that ACE adapts the sharpening strength at the instance level rather than converging to a single task-specific value.

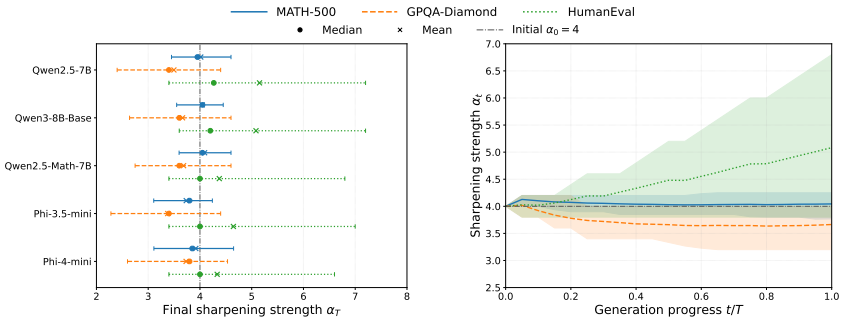


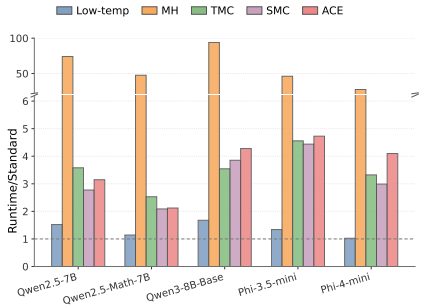
Figure 5: Adaptive sharpening behavior of ACE. Left: Distribution of the final sharpening strength α_T across models and benchmarks. Right: Evolution of the sharpening strength α_t over normalized generation progress for Qwen3-8B-Base. Shaded regions indicate the 25th–75th percentile range.

Figure 5 (right) further shows how the sharpening strength evolves during generation. Starting from the same initialization, α_t stays close to 4 on MATH-500, gradually decreases on GPQA-Diamond, and increases on HumanEval. These distinct trajectories show that ACE dynamically adjusts α_t throughout generation, allowing different instances to follow different sharpening paths as inference proceeds.

6.3 Efficiency and Ablation Studies

Figure 6 (Left) reports the runtime of each method relative to the Standard baseline, averaged over three datasets for each model. MH is substantially more expensive than the other methods, while TMC, SMC, and ACE have comparable computational costs across models. ACE is slightly slower than SMC, with the additional overhead arising from the adaptive updates of α during generation.

As shown in Figure 6 (Right), the ablation study further demonstrates the importance of the entropy-based update mechanism for ACE. Disabling it consistently degrades performance, indicating that high-entropy regions provide a useful signal for adapting the sharpening strength.



Dataset	ACE	ACE w/o Entropy Update
HumanEval	68.3	66.1
GPQA-Diamond	34.4	33.3
MATH-500	68.7	68.3

Figure 6: Left: runtime of different sampling methods across five models. Right: performance of ACE with and without entropy updates on three benchmarks, averaged over five models.

7 Conclusion

In this work, we revisit the role of sharpening strength α in PS and show that stronger sharpening does not necessarily lead to better reasoning performance. Through a preservation–recovery decomposition, we characterize an empirical concentration–exploration trade-off associated with α, helping explain why the appropriate sharpening strength varies across models and tasks. Building on this observation, we propose ACE, a training-free Power Sampling framework that adaptively adjusts α for each problem instance during generation based on the evolving state of the particle population. Experiments across multiple reasoning and code generation benchmarks show that ACE improves over the fixed-α PS baselines in most settings while introducing only modest additional computational cost. Overall, our findings suggest that the sharpening strength in PS should be treated as a dynamic control variable rather than a fixed hyperparameter, enabling inference-time regulation of the concentration–exploration trade-off.

AI USE STATEMENT

Generative AI tools were used to assist with language editing, improving clarity and readability, and providing feedback on the presentation of the manuscript. The authors carefully reviewed and verified all AI-assisted content and take full responsibility for the final manuscript. Generative AI tools were not used to generate experimental results, fabricate data, or replace the authors' scientific judgment.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the proposed method, experimental setup, baseline configurations, and evaluation protocols in the main paper and appendices. The appendices include additional implementation details, hyperparameter settings, model and dataset configurations, ablation studies, and supplementary experimental results. These details are intended to enable independent reproduction of the reported results.

REFERENCES

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024a.

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024b.

Aleksei Arzhantsev, Otmane Sakhi, and Nicolas Chopin. Self-consistency via marginal sharpening. *arXiv preprint arXiv:2605.28142*, 2026.

Seyedarmin Azizi, Erfan Baghaei Potraghloo, Minoo Ahmadi, Souvik Kundu, and Massoud Pedram. Power-smc: Low-latency sequence-level power sampling for training-free llm reasoning. *arXiv preprint arXiv:2602.10273*, 2026.

Kewei Chen, Yayu Long, and Mingsheng Shang. Rethinking instruction drift as a sampling error: SNR-aware power distributions for long-horizon robotic planning. In *Forty-third International Conference on Machine Learning*, 2026a. URL https://openreview.net/forum?id=RKiQ95ZuCW.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Ruishuo Chen, Yu Chen, Zhuoran Li, and Longbo Huang. Powerflow: Unlocking the dual nature of LLMs via principled distribution matching. In *Forty-third International Conference on Machine Learning*, 2026b. URL https://openreview.net/forum?id=m3H7Oexpbx.

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. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. Sequential monte carlo samplers. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 68(3):411–436, 2006.

Mingyuan Fan, Weiguang Han, Daixin Wang, Cen Chen, Zhiqiang Zhang, and Jun Zhou. When sharpening becomes collapse: Sampling bias and semantic coupling in rl with verifiable rewards. *arXiv preprint arXiv:2601.15609*, 2026.

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.

Hong Guo, Nianhui Guo, Christoph Meinel, and Haojin Yang. Sample where you struggle: Sharpening base model reasoning via entropy-guided power sampling. *arXiv preprint arXiv:2606.09926*, 2026.

Andre He, Daniel Fried, and Sean Welleck. Rewarding the unlikely: Lifting GRPO beyond distribution sharpening. *arXiv preprint arXiv:2506.02355*, 2025.

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.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Ajay Jasra, David A Stephens, Arnaud Doucet, and Theodoros Tsagaris. Inference for lévy-driven stochastic volatility models via adaptive sequential monte carlo. *Scandinavian Journal of Statistics*, 38(1):1–22, 2011.

Xiaotong Ji, Rasul Tutunov, Matthieu Zimmer, and Haitham Bou Ammar. Scalable power sampling: Unlocking efficient, training-free reasoning for llms via distribution sharpening. *arXiv preprint arXiv:2601.21590*, 2026.

Aayush Karan and Yilun Du. Reasoning with sampling: Your base model is smarter than you think. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Vsgq2ldr4K.

Xianzhi Li, Ethan Callanan, Abdellah Ghassel, and Xiaodan Zhu. Entropy-gated branching for efficient test-time reasoning. *arXiv preprint arXiv:2503.21961*, 2025.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

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.

Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, et al. Spurious rewards: Rethinking training signals in rlvr. *arXiv preprint arXiv:2506.10947*, 2025.

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.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, et al. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *Advances in Neural Information Processing Systems*, 38:115452–115486, 2026.

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. In *International Conference on Learning Representations*, 2023.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a.

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*, 2024b.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, 2023.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=4OsgYD7em5.

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2026, 2026.

Felix Zhou, Anay Mehrotra, and Quanquan C Liu. Reasoning with sampling: Cutting at decision points. *arXiv preprint arXiv:2605.30327*, 2026.

Yan Zhou, Adam M Johansen, and John AD Aston. Toward automatic model comparison: an adaptive sequential monte carlo approach. *Journal of Computational and Graphical Statistics*, 25 (3):701–726, 2016.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A Proof of Proposition 1

Proof. Fix a generation step t and a particle set $\{y_{1:t}^{(i)}\}_{i=1}^{N}$, and let $p_i = p_\theta(y_{\le t}^{(i)}|x)$, $i = 1, \dots, N$. Under the condition assumed in Proposition 1, the unnormalized weight of particle i under sharpening strength $\alpha \ge 1$ is

$$w_t^{(i)} = w_{t-1}^{(i)} \cdot p_\theta(y_t^{(i)}|y_{<t}^{(i)}, x)^{\alpha-1} = \frac{1}{N} \prod_{s=1}^{t} p_\theta(y_s^{(i)}|y_{<s}^{(i)}, x)^{\alpha-1} = \frac{p_\theta(y_{\le t}^{(i)}|x)^{\alpha-1}}{N}, \tag{7}$$

and the normalized weight is $\bar{w}_i(\alpha) = \frac{p_i^{\alpha-1}}{\sum_{j=1}^{N} p_j^{\alpha-1}}$. Therefore, the effective sample size is

$$\begin{aligned} \mathrm{ESS}_t(\alpha) &= \frac{1}{\sum_{i=1}^{N} \bar{w}_i(\alpha)^2} \\ &= \frac{\left(\sum_{i=1}^{N} p_i^{\alpha-1}\right)^2}{\sum_{i=1}^{N} p_i^{2(\alpha-1)}}. \end{aligned} \tag{8}$$

Let $r = \alpha - 1 \ge 0$ and define the log-partition function $F(r) := \log\left(\sum_{i=1}^{N} p_i^r\right)$. Then

$$\log \mathrm{ESS}_t(\alpha) = 2F(r) - F(2r). \tag{9}$$

Differentiating with respect to r gives

$$\frac{d}{dr}\log \mathrm{ESS}_t(\alpha) = 2F'(r) - 2F'(2r). \tag{10}$$

It remains to show that $F'(r)$ is non-decreasing in r. Differentiating F twice yields

$$F'(r) = \frac{\sum_{i=1}^N p_i^r \log p_i}{\sum_{i=1}^N p_i^r}, \tag{11}$$

and

$$\begin{aligned} F''(r) &= \frac{\sum_{i=1}^N p_i^r (\log p_i)^2}{\sum_{i=1}^N p_i^r} - \left(\frac{\sum_{i=1}^N p_i^r \log p_i}{\sum_{i=1}^N p_i^r}\right)^2 \\ &= \mathrm{Var}_{q_r}\big(\log p_i\big) \geq 0, \end{aligned} \tag{12}$$

where $q_r(i) = \frac{p_i^r}{\sum_{j=1}^N p_j^r}$. Hence, F is convex and $F'(r)$ is non-decreasing. Because $r \geq 0$ implies $2r \geq r$, we have

$$F'(2r) \geq F'(r). \tag{13}$$

Consequently,

$$\frac{d}{dr}\log \mathrm{ESS}_t(\alpha) = 2\big(F'(r) - F'(2r)\big) \leq 0. \tag{14}$$

Since $r = \alpha - 1$ is monotonically increasing in α, $\mathrm{ESS}_t(\alpha)$ is non-increasing with respect to α. Therefore, for any $1 \leq \alpha_1 < \alpha_2$,

$$\mathrm{ESS}_t(\alpha_1) \geq \mathrm{ESS}_t(\alpha_2). \tag{15}$$

Finally, if all particles have identical trajectory probabilities, i.e., $p_1 = p_2 = \cdots = p_N$, then the normalized weights are uniform for every α: $\bar{w}_i(\alpha) = \frac{1}{N}$ and hence $\mathrm{ESS}_t(\alpha) = N$ for all $\alpha \geq 1$. Conversely, if the trajectory probabilities are not all identical, then $\mathrm{Var}_{q_r}(\log p_i) > 0$. F is strictly convex and the ESS strictly decreases as the sharpening strength increases. This establishes the equality condition and completes the proof. □

B Related Work

Sequential Monte Carlo. Sequential Monte Carlo (SMC) methods approximate a sequence of target distributions using a population of weighted particles, which are propagated through importance weighting, resampling, and transition steps (Del Moral et al., 2006). Prior work has also developed adaptive SMC schemes that automatically determine intermediate distributions during sampling. For example, adaptive tempering methods select the next tempering parameter according to statistics of the current particle population, such as the effective sample size (ESS), thereby controlling the change between successive intermediate distributions and mitigating particle degeneracy (Jasra et al., 2011; Zhou et al., 2016).

ACE shares with these methods the general idea of using particle statistics for online adaptation, but addresses a different problem. In conventional adaptive tempering, adaptation determines how to traverse a sequence of intermediate distributions toward a prespecified terminal target, where the target distribution itself is fixed in advance. In contrast, the sharpening strength α in PS directly specifies the trajectory-level distribution from which reasoning trajectories are sampled, and there is no predetermined terminal value of α. ACE therefore allows α_t to increase or decrease throughout autoregressive generation according to the evolving particle population. Rather than adapting a tempering schedule toward a fixed target, ACE dynamically controls the sharpening target itself to regulate concentration and exploration during generation.

Test-Time Scaling for LLM Reasoning. Test-time scaling improves language-model reasoning by allocating additional computation at inference time without modifying model parameters. A major class of approaches increases the breadth of inference by generating multiple candidate solutions and aggregating or selecting among them. Self-consistency samples multiple reasoning traces and aggregates their predictions (Wang et al., 2023), while Best-of-N (Cobbe et al., 2021) and verifier-based methods (Lightman et al., 2023) select promising candidates using model scores, outcome supervision, or process-level verification. Other approaches increase the depth of inference by explicitly exploring intermediate reasoning states through search. For example, Tree of Thoughts (Yao et al., 2023) searches over multiple intermediate reasoning states and evaluates alternative reasoning paths before producing a final solution. Recently, Snell et al. (2024) study adaptive allocation of inference-time computation across problems of varying difficulty to improve the accuracy–compute trade-off.

These approaches primarily scale reasoning through additional generation, candidate selection, or explicit search. ACE explores test-time scaling from a different dimension. Rather than generating multiple trajectories and selecting among them, ACE directly modifies the distribution from which reasoning trajectories are sampled.

Reinforcement Learning as Distribution Sharpening. Reinforcement learning has become an important post-training paradigm for improving the reasoning capabilities of LLMs (Ziegler et al., 2019; Ouyang et al., 2022; Shao et al., 2024; Guo et al., 2025). For reasoning tasks, reinforcement learning with verifiable rewards (RLVR) can substantially improve performance by increasing the probability of sampling successful reasoning trajectories. Recent work analyzed the effect of RLVR from the perspective of distribution sharpening. Yue et al. (2025) show that many successful trajectories produced by RL-trained models already have non-negligible probability under the base-model distribution, suggesting that RL might improve performance only by concentrating probability mass on existing successful trajectories. Similarly, He et al. (2025) find that reinforcement-learning objectives can preferentially amplify already probable successful trajectories while reducing the diversity of sampled reasoning paths. These studies suggest that distribution sharpening can be an important mechanism through which RLVR improves reasoning performance.

Power Sampling. Power Sampling is a training-free approach that sharpens the trajectory-level distribution of a language model by sampling from $\pi_\alpha(y|x) \propto p_\theta(y|x)^\alpha$ with $\alpha \geq 1$. Karan & Du (2026) introduce an MCMC-based sampler for the power distribution and show that sharpening via PS can achieve performance comparable to models post-trained with RL. However, the iterative MCMC procedure incurs substantial inference cost, particularly for long reasoning trajectories.

Following this line of work, subsequent studies improve the sampling efficiency of PS. Ji et al. (2026) estimate the twist function by sampling future continuations, while Azizi et al. (2026) use Sequential Monte Carlo to reduce sampling latency. Moreover, Guo et al. (2026) and Zhou et al. (2026) improve MCMC sampling by concentrating proposals around high-entropy decision points. Other work explores alternative sharpening targets: Arzhantsev et al. (2026) sharpen the marginal answer distribution rather than complete reasoning trajectories, establishing a connection between PS and self-consistency. Beyond LLM reasoning, power distributions have also been applied to long-horizon robotic planning to mitigate instruction drift through inference-time trajectory sharpening (Chen et al., 2026a).

Existing work primarily focuses on how to efficiently sample from a power distribution with a prescribed sharpening strength. In contrast, ACE studies how the sharpening strength itself should be controlled during inference. By revealing a concentration–exploration trade-off governed by α, ACE adaptively adjusts the sharpening strength for each instance throughout generation.

C Experimental Details

C.1 Datasets and Models

Datasets. We evaluate all methods on three reasoning and code generation benchmarks. MATH-500 (Hendrycks et al., 2021) contains 500 competition-level mathematics problems. GPQA-Diamond (Rein et al., 2023) contains 198 multiple-choice questions designed to test expert-level

Table 2: Prompt templates used for the Qwen and Phi model families. For GPQA-Diamond, {A}–{D} denote the four answer choices.

Task	Qwen Prompt	Phi Prompt
MATH-500	`Can you solve the following math problem? {Question} Please reason step by step, and put your final answer within \boxed{}.`	`Can you solve the following math problem? {Question} Please reason step by step, and put your final answer within \boxed{}.`
HumanEval	`{Question}`	`Write a Python function '{signature}' to solve the following problem: {description} {Question}`

GPQA-Diamond	`Answer the following multiple choice question. The last line of your response should be of the following format: '\boxed{$LETTER}' (without quotes) where LETTER is one of ABCD (ex. '\boxed{A}'). Think step by step before answering.`
	
	`{Question}`
	
	`A) {A}`
	`B) {B}`
	`C) {C}`
	`D) {D}`
		`Answer the following multiple choice question. The last line of your response should be of the following format: '\boxed{$LETTER}' (without quotes) where LETTER is one of ABCD (ex. '\boxed{A}'). Think step by step before answering.`
		
		`{Question}`
		
		`A) {A}`
		`B) {B}`
		`C) {C}`
		`D) {D}`

scientific reasoning. HumanEval (Chen et al., 2021) contains 164 Python programming problems and evaluates generated programs using the official functional test cases.

Models. We conduct experiments with five publicly available language models: Qwen2.5-7B (Yang et al., 2024a), Qwen2.5-Math-7B (Yang et al., 2024b), Qwen3-8B-Base (Yang et al., 2025), Phi-3.5-mini (Abdin et al., 2024a), and Phi-4-mini (Abdin et al., 2024b). We use the original pretrained model weights without fine-tuning or an external reward model. All experiments are run with vLLM `0.21.0`, using the model's native tokenizer.

C.2 Prompt Templates

Table 2 reports the prompt templates used throughout our experiments, which are identical to those used in Karan & Du (2026). The same prompt is used by all decoding methods so that performance differences arise from the sampling procedure rather than from prompt engineering. Here, {`Question`} denotes the original problem statement, {`signature`} is the target Python function signature, and {`description`} is the natural-language description extracted from the HumanEval docstring. For GPQA-Diamond, the answer choices are permuted for each example.

C.3 Implementations of Baselines

We compare ACE against two direct autoregressive sampling baselines and three PS baselines. All methods use the same prompts and model weights, and all experiments are run with the vLLM inference backend. We allow at most $3{,}072$ newly generated tokens with a maximum model context length of $4{,}096$ tokens. We report the mean over five independent runs with seeds 42, 1062, 2082, 3102, and 4122. Table 3 summarizes parameters used in the main experiments.

Table 3: Important hyperparameters of the baseline methods. A dash denotes a parameter that is not applicable to the corresponding method.

Method	α / Temp.	Particles / Candidates	Block Size	Key Parameters
Standard	– / 1.0	–	–	–
Low-temp	– / 0.25	–	–	–
MH	4 / 0.25	–	192	10 MH updates
TMC	4 / 0.25	8 candidates	192	8 rollouts, lookahead = 192
SMC	4 / 0.25	64 particles	192	ESS ratio = 0.5

Standard and low-temperature sampling. Standard sampling draws a single autoregressive completion with temperature 1.0 and low-temperature sampling uses temperature 0.25.

Metropolis–Hastings (MH). Following the implementation details in Karan & Du (2026), the MH baseline targets the trajectory-level power distribution with $\alpha = 4$. The proposal distribution is given by the base model with temperature 0.25. Generation proceeds autoregressively in blocks of up to 192 tokens, with 10 MH updates applied at each stage. For each update, a token position is selected uniformly at random, the suffix from that position is resampled from the proposal distribution, and the resulting trajectory is accepted or rejected according to the Metropolis–Hastings acceptance ratio computed from the target and proposal probabilities. Other hyperparameters are set following the official implementation of Karan & Du (2026).

Twisted Monte Carlo (TMC). Following the implementation details in Ji et al. (2026), we implement TMC with sharpening strength $\alpha = 4$ and a block size of 192 tokens. At each block, we consider 8 candidates and estimate the corresponding future-scaling factors using 8 Monte Carlo rollouts per candidate. We use a lookahead horizon of up to 192 tokens and apply the jackknife correction proposed in their estimator. For rollout sampling, we use temperature 0.25.

Sequential Monte Carlo (SMC). Following the implementation details in Azizi et al. (2026), we use $N = 64$ weighted particles and target the trajectory-level power distribution with $\alpha = 4$. We linearly increase the effective exponent from 1 to 4 during the initial stage of generation to reduce early particle degeneracy. Resampling is performed whenever $\mathrm{ESS} < 0.5N$, and the final completion is sampled according to the normalized terminal particle weights. We use a proposal temperature of 0.25, corresponding to $\tau = 1/\alpha$. In the implementation, the block size is 192 and ESS is evaluated at each boundary.

C.4 Implementation of ACE

ACE adapts the sharpening exponent α during generation based on the evolving particle population. Starting from an initial exponent α_0, ACE periodically evaluates a local set of candidate values around the current α_t and selects the largest candidate whose predicted effective sample size (ESS) remains above a prescribed threshold. In addition to these periodic updates, ACE monitors token-level entropy to identify uncertain regions of the generation process and triggers additional updates when a sufficiently large fraction of particles is associated with such regions. This allows the sharpening strength to be adjusted dynamically throughout inference while maintaining particle diversity.

All ACE experiments use $N = 64$ particles and initialize the sharpening exponent at $\alpha_0 = 4$, with α_t constrained to $[1, 10]$. At each α-update step, we construct 16 evenly spaced candidate values within

$$[\max(1, \alpha_t - \Delta_{\mathrm{down}}), \min(10, \alpha_t + \Delta_{\mathrm{up}})]\,, \tag{16}$$

together with the current value α_t. ACE selects the largest candidate whose predicted ESS satisfies the corresponding ESS target in (6). If no candidate satisfies the target, the candidate with the largest predicted ESS is selected. We set $\Delta_{\mathrm{up}} = 0.20$ and $\tau_{\mathrm{mat}} = 0.70$ for all experiments. For HumanEval and GPQA-Diamond, we use $\tau_{\mathrm{ESS}} = 0.50$ and $\Delta_{\mathrm{down}} = 0.20$, while for MATH-500 we use $\tau_{\mathrm{ESS}} = 0.55$ and $\Delta_{\mathrm{down}} = 0.15$. We further examine the sensitivity of ACE to the choice of τ_{ESS} in Appendix D.5.

Algorithm 2 Detailed ACE procedure

Require: Prompt x, LM p_θ, particles N, initial exponent α_0, limit T, block size B, controller settings from Appendix C.4.

Initialize empty prefixes, weights $w_0^{(i)} = 1/N$, and uncertainty states.
for $t = 1, \ldots, T$ **do**
  Set $\alpha_t \leftarrow \alpha_{t-1}$.
  for each unfinished particle i in parallel **do**
    Generate $y_t^{(i)}$ by low-temperature sampling from p_θ at temperature $1/\alpha_t$.
    Append the token to $y_{<t}^{(i)}$ and update $w_t^{(i)}$ using Equation (4).
    Mark a particle as an uncertainty candidate when its base-model token entropy exceeds 1.5 and mark the candidate mature after it generates another L tokens.
  end for
  if a block boundary or uncertainty trigger is reached **then**
    Construct $\mathcal{A}_t$ using the candidate bounds and grid specified above.
    for each $\alpha_c \in \mathcal{A}_t$ **do**
      Compute $w_t^{(i)}(\alpha_c)$ using Equation (5) and evaluate $\mathrm{ESS}_t(\alpha_c)$.
    end for
    Select the largest α_c satisfying $\mathrm{ESS}_t(\alpha_c) \geq \tau_{\mathrm{ESS}} N$.
    If none satisfies the constraint, select a candidate maximizing $\mathrm{ESS}_t(\alpha_c)$.
    Set $w_t^{(i)} \leftarrow w_t^{(i)}(\alpha_c)$ for all particles and $\alpha_t \leftarrow \alpha_c$.
    Recompute ESS_t using the updated weights.
    if $\mathrm{ESS}_t < 0.5N$ **then**
      Resample particles according to their normalized weights.
      Copy their prefixes, cumulative log-probabilities, caches, and uncertainty states.
      Copy termination flags and reset $w_t^{(i)} \leftarrow 1/N$ for all particles.
    end if
  end if
  if this step included an uncertainty-triggered update **then**
    Reset uncertainty candidate flags and maturity states for all particles.
  end if
  if all particles have finished **then**
    break
  end if
end for
Sample a particle index i with probability $w_t^{(i)} / \sum_{j=1}^{N} w_t^{(j)}$.
return the selected completion $y^{(i)}$.

Periodic α updates and ESS-based resampling checks are performed every 192 generated tokens. ACE uses an adaptive proposal temperature of $1/\alpha_t$. Resampling is performed whenever the current ESS falls below $0.5N$. The final output is sampled according to the normalized terminal particle weights.

For the entropy-based update mechanism, ACE monitors the entropy of the base-model next-token distribution for each particle. An uncertainty window is initiated when the entropy exceeds 1.5. After a lookahead of $L = 8$ tokens, the corresponding particle contributes to the uncertainty-triggering statistic. An additional α update is triggered once the number of mature particles exceeds a prescribed threshold $\tau_{\mathrm{mat}} N$. ESS is then re-evaluated and resampling is performed when necessary. After each uncertainty-triggered update and any resulting resampling, the uncertainty candidate flags and maturity states of all particles are reset. Each unfinished particle can then be marked again upon encountering a new high-entropy token and becomes mature after another L tokens, allowing uncertainty detection to continue throughout generation.

Detailed algorithm. Algorithm 2 expands the main procedure using the proposal and controller settings above. Candidate ESS values are evaluated before resampling. After the selected exponent is applied, ESS is recomputed to determine whether resampling is needed. Resampled particles inherit their ancestors' prefixes, cumulative base-model log-probabilities, decoding caches, and uncertainty states; only their importance weights are reset.

C.5 Diagnostic Experiments on Sharpening Strength in Section 4

The experiments in Section 4 use Power Sampling implemented with Sequential Monte Carlo (SMC) (Del Moral et al., 2006; Azizi et al., 2026). We evaluate the five models listed in Section 4 on MATH-500, GPQA-Diamond, and HumanEval, sweeping the fixed sharpening strength over $\alpha \in \{2, 3, \ldots, 10\}$. For each model, benchmark and α, we conduct three runs using seeds 42, 1062, and 2082 and report their average results. For the preservation–recovery analysis, we use the outcomes at $\alpha = 10$ as the reference and evaluate the other exponents on the corresponding fixed subsets, following the definitions in Section 4. The metrics are computed for each seed before averaging across seeds.

D ADDITIONAL EXPERIMENTS

D.1 ADDITIONAL EXPERIMENTS OF SECTION 4

The diagnostic studies in Section 4 are conducted with PS implemented using SMC (Del Moral et al., 2006; Azizi et al., 2026). To assess the robustness of these findings to the choice of sampling method, we repeat the sharpening-strength sweep and preservation–recovery analysis using TMC (Ji et al., 2026) and examine whether similar patterns arise under a different sampling scheme. We evaluate Qwen3-8B-Base and Phi-4-mini on MATH-500, GPQA-Diamond, and HumanEval with $\alpha \in \{2, 3, \ldots, 10\}$. All results are averaged over three random seeds. For the preservation–recovery analysis, we use $\alpha = 10$ as the reference setting, following Section 4.

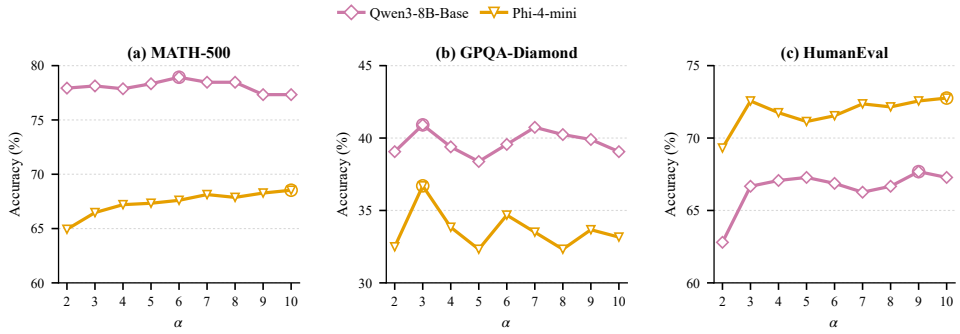


Figure 7: Reasoning accuracy under TMC across different sharpening strengths α for Qwen3-8B-Base and Phi-4-mini on three benchmarks. Curves report the mean over three random seeds.

Effect of sharpening strength. Figure 7 shows that increasing the sharpening strength does not consistently improve reasoning accuracy under TMC, exhibiting the same qualitative pattern as results in Figure 2. On MATH-500, Qwen3-8B-Base attains its highest observed accuracy at $\alpha = 6$ and degrades under stronger sharpening, whereas Phi-4-mini generally improves with α and reaches its highest observed accuracy at $\alpha = 10$. On GPQA-Diamond, both models achieve their highest observed accuracy at $\alpha = 3$, followed by non-monotonic fluctuations as α increases. On HumanEval, Qwen3-8B-Base reaches its highest observed accuracy at $\alpha = 9$, while Phi-4-mini performs best at $\alpha = 10$, with both curves exhibiting local reversals over the sweep. Overall, the best-performing sharpening strength depends on the model–benchmark pair, and no single value of α is uniformly optimal across these TMC experiments.

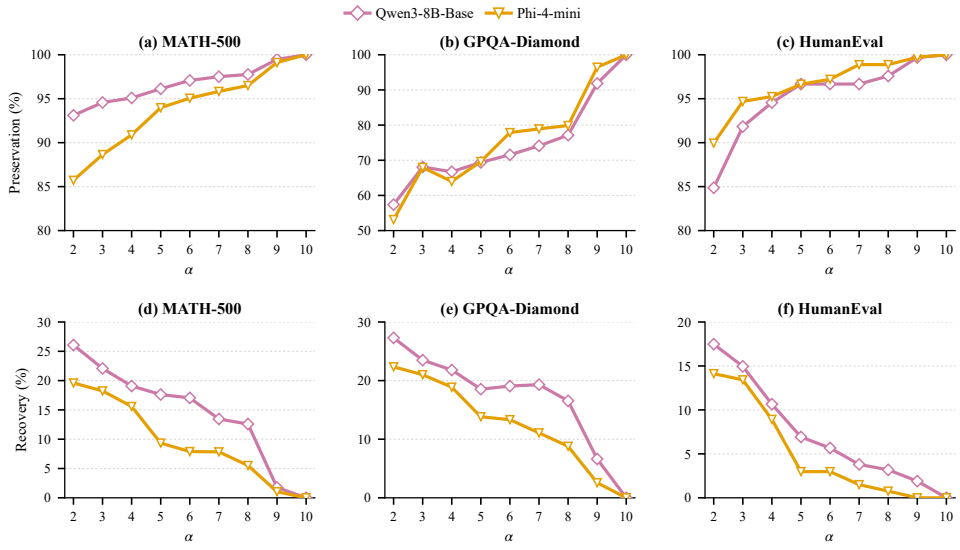


Figure 8: Preservation and recovery of TMC under the partition defined by the outcomes of $\alpha = 10$.

Preservation and recovery. Figure 8 exhibits a similar pattern to Figure 3: preservation generally increases with α, whereas recovery generally decreases. Across the three benchmarks, stronger sharpening increasingly preserves solutions obtained under the reference setting, while weaker sharpening retains a greater ability to recover solutions missed under strong sharpening. Similar to the analysis in Section 4, this opposing behavior of preservation and recovery helps explain the non-monotonic accuracy trends in Figure 7: increasing α can improve preservation while simultaneously reducing recovery, so stronger sharpening does not necessarily yield higher overall accuracy. Taken together, these results reproduce the qualitative conclusion of Section 4 under TMC, suggesting that the sharpening strength α governs a similar concentration–exploration trade-off in PS.

Discussion of the preservation–recovery decomposition. The preservation–recovery analysis uses the outcomes at $\alpha = 10$ to define the fixed subsets $\mathcal{D}^{+}_{10}$ and $\mathcal{D}^{-}_{10}$. Consequently, $\mathrm{Preservation}(10) = 1$ and $\mathrm{Recovery}(10) = 0$ by construction, provided the corresponding subsets are nonempty. Importantly, however, these endpoint values do not determine the behavior at the remaining sharpening strengths. Once the reference partition is fixed, preservation and recovery at each $\alpha < 10$ are evaluated independently on the same subsets, and the definitions impose neither monotonicity nor any particular rate of change across the sweep.

Therefore, the evidence in Figures 3 and 8 comes from how preservation and recovery vary over the non-reference exponents, rather than from the prescribed values at $\alpha = 10$ itself. The observed tendency for preservation to increase and recovery to decrease as sharpening becomes stronger is thus not a direct consequence of the endpoint definition. Instead, the reference partition provides a common basis for separating two complementary changes in instance-level correctness: retaining solutions already obtained under strong sharpening and recovering solutions that strong sharpening misses.

This decomposition also provides a descriptive explanation for the non-monotonic accuracy trends. With $\rho = |\mathcal{D}^{+}_{10}|/|\mathcal{D}|$,

$$\mathrm{Perf}(\alpha) = \rho\,\mathrm{Preservation}(\alpha) + (1-\rho)\,\mathrm{Recovery}(\alpha).$$

Thus, improvements in preservation can be offset by reductions in recovery. Importantly, this identity does not itself imply either trend. Instead, it provides a diagnostic decomposition of overall accuracy that helps characterize how the opposing preservation and recovery effects vary with the sharpening strength.

D.2 Comparison with Tuned Fixed-α SMC

The main experiments in Section 6 compare ACE with SMC using the commonly adopted fixed sharpening strength $\alpha = 4$. To further examine whether the advantage of adaptive sharpening persists when the fixed exponent is tuned for each model–benchmark pair, we conduct an additional comparison with tuned fixed-α SMC. We consider Qwen3-8B-Base and Phi-4-mini on MATH-500, GPQA-Diamond, and HumanEval.

Selection of fixed sharpening strengths. For each model and benchmark, we sweep $\alpha \in \{2, 3, \ldots, 10\}$ using seed 42 and select the exponent that achieves the highest accuracy:

$$\alpha^{\star} \in \operatorname*{arg\,max}_{\alpha \in \{2,\ldots,10\}} \mathrm{Acc}(\alpha).$$

The resulting $\alpha^{\star}$ is then used as a fixed hyperparameter for the corresponding model–benchmark pair in the subsequent evaluation. Unlike ACE, which adapts α online during generation, the tuned baseline determines $\alpha^{\star}$ in advance through this offline sweep and keeps it unchanged across all problem instances and generation steps.

As shown in Table 4, the selected α varies substantially across model–benchmark pairs. For Qwen3-8B-Base, GPQA-Diamond favors weak sharpening with $\alpha^{\star} = 2$, whereas HumanEval selects a substantially larger value of $\alpha^{\star} = 8$. For Phi-4-mini, the selected exponent ranges from 3 on HumanEval to 6 on MATH-500. These results further illustrate that a single fixed exponent is unlikely to be uniformly suitable across different models and tasks.

Table 4: Best-performing fixed sharpening strength $\alpha^\star$ for SMC on seed 42, selected separately for each model–benchmark pair from the sweep over $\alpha \in \{2, \ldots, 10\}$.

Model	Benchmark	Selected $\alpha^\star$
Qwen3-8B-Base	MATH-500	6
	GPQA-Diamond	2
	HumanEval	8
Phi-4-mini	MATH-500	6
	GPQA-Diamond	5
	HumanEval	3

Table 5: Mean accuracy (%) over five runs for SMC with $\alpha = 4$, tuned SMC with $\alpha^\star$, and ACE. The selected $\alpha^\star$ values are reported in Table 4. Avg. denotes the mean across the six model–benchmark settings. Bold indicates the best mean accuracy in each setting.

	Qwen3-8B-Base			Phi-4-mini			Avg.
Method	HumanEval	GPQA-Diamond	MATH-500	HumanEval	GPQA-Diamond	MATH-500	
SMC ($\alpha = 4$)	67.7	37.4	79.4	76.7	29.2	63.0	58.9
SMC (tuned)	66.3	**43.3**	79.3	75.4	30.1	65.6	60.0
ACE	**70.0**	41.0	**79.5**	**77.6**	**30.6**	**66.2**	**60.8**

Comparison with tuned SMC. We next evaluate the selected fixed-α SMC configurations over five runs and compare them with the default SMC baseline using $\alpha = 4$ and with ACE.

Table 5 shows that tuning the sharpening strength can improve SMC performance in several settings, although the gains are not uniform. The largest improvement occurs for Qwen3-8B-Base on GPQA-Diamond, where tuning increases accuracy from 37.4% to 43.3%. For Phi-4-mini, tuning also improves accuracy on MATH-500 from 63.0% to 65.6% and on GPQA-Diamond from 29.2% to 30.1%. In contrast, tuning does not improve the performance on HumanEval for either model. Since $\alpha^\star$ is selected based on the performance of a single seed, the chosen exponent may not correspond to the best average performance across subsequent runs. Averaged across the six model–benchmark settings, tuning increases the mean accuracy from 58.9% to 60.0%.

Compared with tuned fixed-α SMC, ACE achieves higher accuracy in five of the six model–benchmark settings. Averaged across all six settings, ACE reaches 60.8% accuracy, compared with 60.0% for tuned SMC. These results show that model- and benchmark-specific tuning can strengthen the fixed-α SMC baseline, while ACE retains a higher average accuracy even after such tuning. Unlike tuned SMC, which requires an offline search to determine a fixed exponent for each model–benchmark pair, ACE adapts the sharpening strength online according to the evolving particle population during generation. This comparison further supports the benefit of adaptive sharpening and indicates that ACE's gains are not merely due to using $\alpha = 4$ as the fixed-α baseline.

D.3 Additional Results on AIME 2026

We additionally evaluate all six methods on AIME 2026 (Zhang & Math-AI, 2026), which contains 30 competition-level mathematics problems. We use the same five models, single-shot evaluation protocol, and five random seeds $\{42, 1062, 2082, 3102, 4122\}$ as in the main experiments. The prompt follows the MATH-500 template in Appendix C.2, while the generation budget and baseline configurations follow Appendix C.3. For ACE, we use the same controller configuration as on MATH-500, with $\alpha_0 = 4$, $\tau_{\mathrm{ESS}} = 0.55$, $\Delta_{\mathrm{up}} = 0.20$, $\Delta_{\mathrm{down}} = 0.15$, and $\tau_{\mathrm{mat}} = 0.70$.

Results. As shown in Table 6, ACE achieves the best performance on Qwen3-8B-Base and Phi-3.5-mini, and the second-best performance on both Qwen2.5 models. Although ACE is less competitive on Phi-4-mini, it achieves the highest average accuracy across all five models.

Averaged across all five models, ACE attains a mean accuracy of 9.1%, outperforming SMC at 8.7% and the other evaluated baselines. Overall, these results further support the effectiveness of adaptive sharpening on AIME 2026.

Table 6: Accuracy (%) on AIME 2026, averaged over five random seeds. Bold indicates the best result for each model, and underlined values indicate the second-best result. The final row reports the mean accuracy across the five models.

Model	Standard	Low-temp	MH	TMC	SMC	ACE
Qwen2.5-7B	2.7	4.0	8.0	5.3	**10.7**	<u>9.3</u>
Qwen2.5-Math-7B	1.3	8.0	8.7	9.3	**14.0**	<u>12.0</u>
Qwen3-8B-Base	5.3	7.3	8.7	8.7	<u>10.7</u>	**15.3**
Phi-3.5-mini	0.0	0.7	0.7	<u>2.0</u>	1.3	**2.7**
Phi-4-mini	3.3	7.3	<u>8.7</u>	**10.0**	6.7	6.0
Avg.	2.5	5.5	6.9	7.1	<u>8.7</u>	**9.1**

D.4 Effects of Adaptive Sharpening Strength and Proposal Temperature

As described in Appendix C.4, the implementation of ACE adapts the trajectory-level sharpening strength α_t during generation and simultaneously sets the proposal temperature to $1/\alpha_t$. To disentangle the effects of these two forms of adaptation, we consider two ablated variants: α**-only** and **temperature-only**. In α**-only**, the trajectory-level sharpening exponent is adapted using the same ACE controller, while the proposal temperature is fixed at 0.25. In **temperature-only**, the trajectory-level sharpening strength is fixed at $\alpha = 4$, while only the temperature is adjusted according to the adaptive value produced by the controller.

We evaluate Qwen3-8B-Base and Phi-4-mini on MATH-500, GPQA-Diamond, and HumanEval. For each configuration, we report the mean accuracy over five independent runs.

Table 7: Accuracy (%) for ACE and variants that adapt only the trajectory-level sharpening exponent or only the proposal temperature. Results are reported as mean over five independent runs.

	Qwen3-8B-Base			Phi-4-mini		
Method	HumanEval	GPQA-Diamond	MATH-500	HumanEval	GPQA-Diamond	MATH-500
ACE	70.0	41.0	79.5	77.6	30.6	66.2
α-only	69.6	40.7	79.4	77.3	33.6	66.2
Temperature-only	69.8	40.0	78.8	76.5	30.1	65.9

Results. As shown in Table 7, adapting the trajectory-level sharpening strength alone largely preserves the performance of full ACE. The α-only variant achieves similar accuracy to ACE in five of the six model–benchmark settings, with only small differences on HumanEval and MATH-500. On Phi-4-mini with GPQA-Diamond, however, α-only achieves a higher mean accuracy than full ACE.

By contrast, adapting only the proposal temperature consistently yields lower accuracy than full ACE across all six settings. These results suggest that the benefit of ACE is primarily associated with online adaptation of the trajectory-level sharpening strength, rather than with adaptive proposal temperature alone. Overall, changing the temperature appears to have a comparatively smaller effect on final accuracy in the evaluated settings.

D.5 Additional Ablation Studies

We further study the sensitivity of ACE to two components of the adaptive controller: the ESS target τ_{ESS} and the initial sharpening exponent α_0. Experiments are conducted on Qwen3-8B-Base and Phi-4-mini using HumanEval and MATH-500. For each configuration, we report the mean accuracy over three independent runs.

Sensitivity to the Adaptive ESS Target. We vary τ_{ESS} from 0.1 to 1.0 in increments of 0.1, while keeping the remaining settings unchanged. The ESS target determines the minimum predicted ESS required for a candidate sharpening strength to be accepted by the adaptive controller.

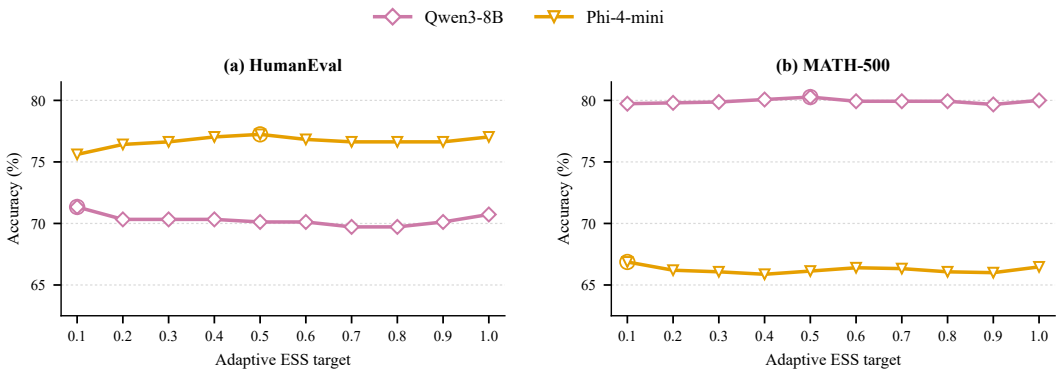


Figure 9: Sensitivity of ACE accuracy to the adaptive ESS target τ_{ESS} on (a) HumanEval and (b) MATH-500. Results are averaged over three independent runs.

As shown in Figure 9, ACE is relatively insensitive to the ESS target over the evaluated range. For both models, accuracy remains within a narrow range on MATH-500, and only moderate variation is observed on HumanEval. Moreover, there is no consistent increasing or decreasing trend as τ_{ESS} changes. These results suggest that the ESS constraint mainly provides a stable criterion for regulating particle concentration, without requiring a highly precise choice of its target value.

Sensitivity to the Initial Sharpening Exponent. We next vary the initial sharpening exponent over $\alpha_0 \in \{2, 3, \ldots, 10\}$. Unlike the ESS target, α_0 directly determines the sharpening strength at the beginning of generation, after which ACE progressively adapts the exponent according to the evolving particle population.

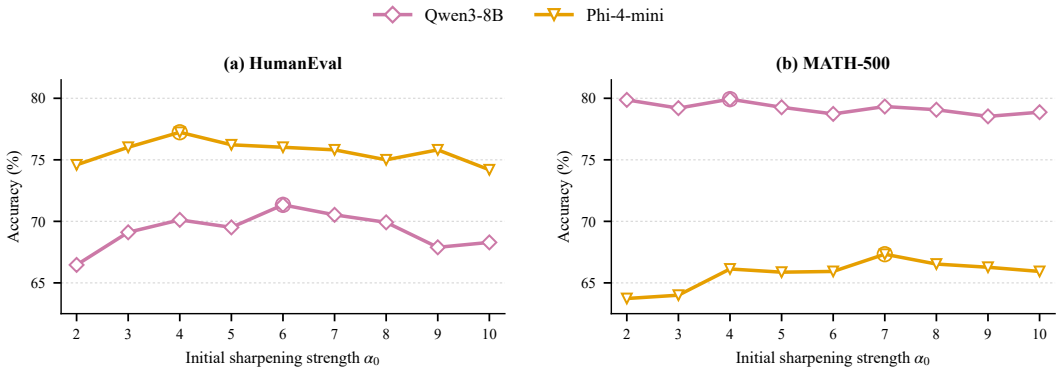


Figure 10: Sensitivity of ACE accuracy to the initial sharpening exponent α_0 on (a) HumanEval and (b) MATH-500. Results are averaged over three independent runs.

Figure 10 shows a stronger dependence on initialization, particularly on HumanEval. Different initial exponents can lead to noticeably different final accuracies, whereas the variation on MATH-500 is comparatively smaller. This sensitivity is natural because α_0 affects the particle distribution from the beginning of generation: different initial sharpening strengths can emphasize different early trajectories, which in turn changes the particle population on which subsequent adaptive updates operate. Since ACE updates α progressively from its current value rather than re-optimizing the exponent globally at each step, the effect of the initialization can persist as generation proceeds.

These results indicate that online adaptation reduces, but does not completely eliminate the influence of the initial sharpening strength. At the same time, the common initialization $\alpha_0 = 4$ used throughout the main experiments provides an effective starting point across the evaluated models and benchmarks. Therefore, although further improvements may be possible through a better choice of initialization, ACE does not require model- or benchmark-specific initialization tuning to achieve strong overall performance.

This sensitivity is also consistent with the diagnostic results in Section 4, where the preferred sharpening strength varies across models and benchmarks, as well as with the instance-level adaptation.

The current ACE framework primarily focuses on adapting the sharpening strength during generation from a common initialization. The observed sensitivity to α_0 therefore suggests a complementary direction for future work: developing mechanisms that adapt or estimate the initial sharpening strength before substantial trajectory generation has occurred. More generally, jointly determining an appropriate initialization and subsequently adapting α during inference may provide a more flexible mechanism for regulating the concentration–exploration trade-off.

D.6 Sharpening Trajectories Across Models

We further extend the analysis in Figure 5 (right), which illustrates the evolution of α_t for Qwen3-8B-Base, to four additional models across the same set of benchmarks. All models start from the same initialization $\alpha_0 = 4$, and each curve tracks α_t over normalized generation progress on MATH-500, GPQA-Diamond, and HumanEval.

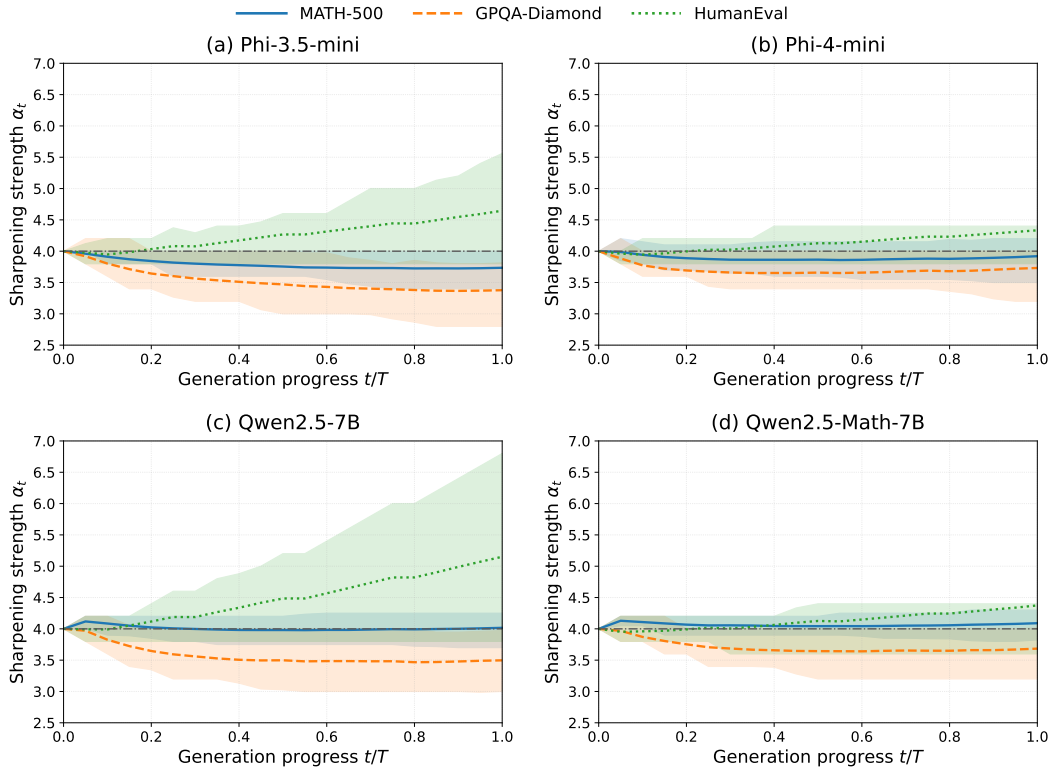


Figure 11: Evolution of the adaptive sharpening exponent α_t over normalized generation progress t/T for four additional models. Curves correspond to MATH-500, GPQA-Diamond, and HumanEval. The horizontal reference line denotes the common initialization $\alpha_0 = 4$.

Across the four additional models, the evolution of α_t exhibits clear variation across both tasks and models. On HumanEval, α_t generally increases over generation, with the most pronounced rise observed for Qwen2.5-7B, whereas on GPQA-Diamond it consistently decreases below the initialization. The trajectories on MATH-500 remain comparatively close to $\alpha_0 = 4$, with only modest adjustments for most models.

Several trajectories are also non-monotonic, indicating that the adaptation does not simply follow a predetermined increasing or decreasing schedule. Together with Figure 5, these results show that ACE dynamically adjusts the sharpening strength in response to the evolving particle population, allowing different models and tasks to follow distinct adaptation trajectories throughout generation.

D.7 Main Results with Standard Deviations

Table 8 reproduces Table 1 with standard deviations across five random seeds.

Table 8: Accuracy (%) reported as mean ± standard deviation over five random seeds. Bold values indicate the highest accuracy for each model and benchmark. Avg. reports the unweighted mean across the three main benchmarks, computed before rounding.

Model	Method	HumanEval	GPQA-Diamond	MATH-500	Avg.
Qwen2.5-7B	Standard	29.3 ± 4.0	28.3 ± 3.2	39.7 ± 2.0	32.4
	Low-temp	56.0 ± 2.7	29.7 ± 2.6	63.7 ± 1.4	49.8
	MH	57.2 ± 1.6	31.4 ± 1.6	69.5 ± 1.6	52.7
	TMC	54.2 ± 1.7	32.0 ± 1.9	62.9 ± 2.7	49.7
	SMC	58.1 ± 1.5	32.1 ± 2.9	**72.5** ± 1.4	54.2
	ACE	**62.3** ± 0.8	**33.1** ± 2.2	71.2 ± 0.3	**55.5**
Qwen2.5-Math-7B	Standard	28.2 ± 1.7	30.5 ± 2.9	48.4 ± 1.8	35.7
	Low-temp	51.8 ± 2.3	35.2 ± 2.4	66.3 ± 1.6	51.1
	MH	55.4 ± 1.5	35.7 ± 1.3	71.4 ± 0.8	54.2
	TMC	52.0 ± 3.2	**36.1** ± 2.8	68.0 ± 2.8	52.0
	SMC	55.0 ± 1.6	33.4 ± 3.1	76.1 ± 0.8	54.9
	ACE	**57.4** ± 1.1	33.6 ± 2.0	**77.1** ± 0.9	**56.1**
Qwen3-8B-Base	Standard	42.6 ± 2.3	34.3 ± 2.3	59.4 ± 1.9	45.4
	Low-temp	65.6 ± 2.3	40.4 ± 3.3	77.7 ± 0.8	61.2
	MH	67.3 ± 1.9	40.7 ± 0.8	**79.8** ± 0.7	62.6
	TMC	67.1 ± 2.0	39.3 ± 1.6	78.0 ± 0.6	61.5
	SMC	67.7 ± 2.0	37.4 ± 0.7	79.4 ± 0.8	61.5
	ACE	**70.0** ± 0.8	**41.0** ± 1.8	79.5 ± 0.8	**63.5**
Phi-3.5-mini	Standard	58.7 ± 2.7	33.1 ± 1.9	42.0 ± 1.3	44.6
	Low-temp	70.2 ± 1.5	33.5 ± 2.4	45.8 ± 1.1	49.8
	MH	71.2 ± 0.8	33.9 ± 3.0	48.6 ± 0.7	51.3
	TMC	69.4 ± 2.1	**35.2** ± 4.3	45.1 ± 1.3	49.9
	SMC	**74.4** ± 0.9	32.3 ± 1.2	49.3 ± 1.5	52.0
	ACE	74.3 ± 1.3	33.4 ± 1.8	**49.7** ± 1.1	**52.5**
Phi-4-mini	Standard	50.9 ± 2.0	30.6 ± 1.8	51.8 ± 1.6	44.4
	Low-temp	70.5 ± 1.6	32.9 ± 3.3	66.0 ± 1.2	56.5
	MH	74.0 ± 1.9	**34.2** ± 2.2	**68.4** ± 0.8	**58.9**
	TMC	71.8 ± 0.3	34.0 ± 2.4	67.0 ± 1.4	57.6
	SMC	76.7 ± 1.2	29.2 ± 1.4	63.0 ± 0.8	56.3
	ACE	**77.6** ± 0.8	30.6 ± 4.0	66.2 ± 0.8	58.1