From Noise to Diversity: Random Embedding Injection in LLM Reasoning

Heejun Kim[1,*] **Seungpil Lee**[1,*] **Jewon Yeom**[2] **Jaewon Sok**[2]
Seonghyeon Park[2] **Jeongjae Park**[2] **Taesup Kim**[2,†] **Sundong Kim**[1,†]

[1]Gwangju Institute of Science and Technology [2]Seoul National University

Abstract

Recent *soft prompt* research has tried to improve reasoning by inserting trained vectors into LLM inputs, yet whether the gain comes from the *learned content* or from the *act of injection itself* has not been carefully separated. We study Random Soft Prompts (RSPs), which drop the training step entirely and append a freshly drawn sequence of random embedding vectors to the input. Each RSP vector is sampled from an isotropic Gaussian fitted to the entrywise mean and variance of the pretrained embedding table. The sequence carries no learned content, yet reaches accuracy comparable to optimized *soft prompts* on math reasoning benchmarks in several settings. The mechanism unfolds in two stages: because attention has to absorb a never-seen-before random position, the distribution over the first few generated tokens flattens and reasoning trajectories *branch*, and as generation continues this influence dilutes naturally so the response commits to a single completion. We show that during inference RSPs lift early-stage token diversity and, combined with temperature sampling, widen Pass@N, the probability that at least one out of N attempts is correct. Beyond inference, RSP collects correct solutions for more hard problems and improves subsequent supervised fine-tuning. Our contributions are: (i) RSP isolates the simplest form of *soft prompt*, which is training-free and freshly resampled, providing a unified lens for the structural effect of injection that variants otherwise differing in training and form all share, (ii) a theoretical and empirical validation of the underlying mechanism, and (iii) an application of RSP to training-data collection that improves subsequent supervised fine-tuning without RSP at inference.

Project page: `heejunkim00.github.io/projects/RSP`

1 Introduction

As LLMs scale to billions of parameters, full fine-tuning is prohibitive, motivating parameter-efficient methods [Hu et al., 2022, Houlsby et al., 2019]. Among these, *soft prompts* insert learnable continuous embeddings into the input and steer behavior through attention [Li and Liang, 2021, Lester et al., 2021]. This paradigm is actively adopted for mathematical reasoning [Kang et al., 2026, Xu et al., 2025, Ye et al., 2026, Hao et al., 2025, Zhang et al., 2026]. These methods diverge in injection position, training objective, and prompt form, and prior work attributes their gains to the *learned content* of the optimized vectors. The act of injecting embeddings into the input, common to all variants, has remained outside the analysis. Even though theory shows trained prefixes only bias attention in a fixed direction [Petrov et al., 2024], the source of the effect itself remains unsettled.

To separate *learned content* from the *act of injection*, we use Random Soft Prompts (RSPs) as a control: continuous vectors drawn from an isotropic Gaussian fitted to the embedding table's mean

[*]Equal contribution. [†]Co-corresponding authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

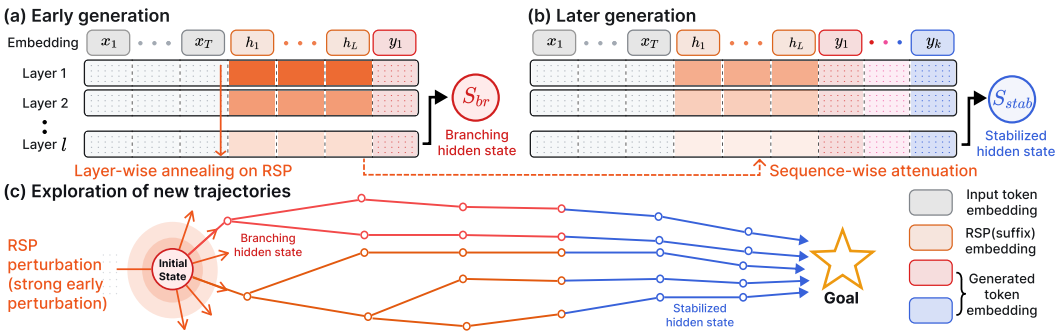


Figure 1: **Conceptual overview of RSP-induced trajectory diversity.** The hidden states shown are the final-layer outputs that drive the next-token distribution. (a) Early decoding: RSP perturbation reaches a *branching state* (red, formalized in §3.1), whose top-K differs from the no-RSP baseline and induces a diverse output distribution. (b) Later decoding: As generation proceeds, RSP attention mass decreases and the hidden state (blue) stabilizes along the branch opened earlier. (c) Across rollouts, the trajectories opened by early RSP perturbation accumulate into independent paths.

and variance, with no learning. The control turns out to be non-neutral. Applying a chat template to a base model not fine-tuned for it degrades reasoning, yet *prefix* RSPs recover $+27.16$ and $+28.62$ pp on MATH-500 and GSM8K, respectively, for Qwen2.5-Math-1.5B (Table 2). The injection itself, independent of any learned content, alters model behavior. To understand how injection alone produces these effects, we examine the resulting generation dynamics. With each rollout receiving an *independent* draw, a single RSP injection *branches* hidden states into diverse trajectories early on, which then *stabilize* as decoding proceeds. Empirically, RSPs share the early-decoding entropy signature of optimized methods (LTPO, TTSV, SoftCoT) despite carrying no learned content. Independent RSP resampling improves Pass@N. RSP also collects correct solutions for more hard problems, improving subsequent supervised fine-tuning without RSP at inference.

Our contributions are: (i) RSP isolates the simplest form of *soft prompt*, training-free and freshly resampled per rollout, providing a unified lens for the structural effect of injection across variants that otherwise differ in training objective and prompt form, while still reaching accuracy comparable to optimized methods in several settings. (ii) We provide a theoretical and empirical account of the mechanism: attention-mediated branch opening, isotropic coverage across all directions, and the observed attenuation of RSP attention explained through token counts and the evolving attention-logit gap. (iii) We apply RSP to rejection-sampling SFT, showing that correct solutions collected with RSP improve downstream accuracy without RSP at inference.

2 Background

2.1 Soft Prompt Methods for LLM Reasoning

Prior *soft prompt* methods share a common structure: they inject continuous vectors that do not correspond to discrete vocabulary tokens into the model's representation stream and *optimize* them under task-specific objectives. As parameter-efficient alternatives to full fine-tuning, Prefix-Tuning [Li and Liang, 2021], Prompt Tuning [Lester et al., 2021], and P-tuning [Liu et al., 2024] reached near-fine-tuning performance using learnable continuous vectors, after which LLM-reasoning variants followed. TTSV [Kang et al., 2026] optimizes *prefix* embeddings at test time via trajectory entropy minimization. SoftCoT [Xu et al., 2025] replaces explicit chain-of-thought tokens with soft tokens generated by an assistant model. LTPO [Ye et al., 2026] optimizes the *soft prompt* per problem to maximize its own confidence on that problem. COCONUT [Hao et al., 2025] feeds its hidden states back as input embeddings. MemGen [Zhang et al., 2026] uses embeddings as experience memory. Pause tokens [Goyal et al., 2024] insert learnable tokens to provide additional computation steps.

These methods differ in position, training stage, and training target, yet assume that *learned representations* are the core driver of the effect. This paradigm faces several limitations. **(i)** Training is domain/benchmark-specific, requiring re-optimization for new settings and sensitive to seed/hyperparameter choices. **(ii)** Several have been observed to degrade on out-of-distribution challenging

Table 1: **Comparison with optimized *soft prompt* methods.** Accuracy (%) under unified evaluation, reported as mean ± std over five initialization seeds for RSP, TTSV, SoftCoT, and LTPO. The baseline uses greedy decoding. RSP uses *suffix* injection, with baseline and RSP values matching Table 2. **Bold** and underline denote the highest and second-highest mean accuracy in each row, respectively.

Model	Benchmark	Baseline	RSP (*suffix*)	TTSV	SoftCoT	LTPO
Qwen2.5-Math-7B-Instruct	MATH-500	83.20	**83.52** ±0.56	83.44 ±0.55	82.88 ±0.32	82.64 ±0.43
	GSM8K	95.45	**95.60** ±0.08	95.12 ±0.11	94.74 ±0.28	94.97 ±0.11
	AIME24	13.33	16.67 ±0.00	15.32 ±4.11	**18.00** ±2.67	12.67 ±1.33
Qwen2.5-Math-1.5B-Instruct	MATH-500	73.00	**74.12** ±1.36	72.32 ±1.75	72.88 ±2.27	73.68 ±0.97
	GSM8K	**85.29**	84.99 ±0.22	84.04 ±1.07	82.97 ±2.97	84.76 ±0.52
	AIME24	10.00	**13.33** ±5.27	6.02 ±1.22	9.33 ±2.42	9.33 ±1.33

tasks [Ye et al., 2026]. **(iii)** Evidence for the effect is largely empirical, with limited theoretical explanation beyond the training objective. Furthermore, soft prompting and prefix-tuning are theoretically restricted to biasing attention outputs in a fixed direction, without changing the relative attention pattern among content tokens [Petrov et al., 2024]. Together, these findings cast doubt on whether the learned prompt vectors function as intended: they are suitable for *eliciting* or *combining* skills in the pretrained model, but struggle to learn new capabilities. These studies leave the contribution of embedding injection itself insufficiently distinguished from that of learned content. We investigate this distinction through RSP and controlled interventions on trained prompt vectors.

2.2 Noise Injection in Neural Networks

Neural networks have incorporated noise at several stages and locations. During training, dropout [Srivastava et al., 2014] randomly deactivates *hidden activations*, and NEFTune [Jain et al., 2024] *adds* uniform noise to *input embeddings* during instruction fine-tuning to improve instruction following. During inference, randomized smoothing [Cohen et al., 2019] injects Gaussian noise into *raw inputs* to obtain certified robustness, and SmoothLLM [Robey et al., 2025] perturbs prompts at the *discrete character* level to defend against jailbreaking. In RL, NoisyNet [Fortunato et al., 2018] *adds* learnable noise to *network weights* to aid exploration. Rather than *adding* noise to embeddings, RSP *appends* it, and differs in two respects: **(i)** unlike NoisyNet, it requires neither model fine-tuning nor learned noise parameters. **(ii)** RSP does not require aggregating predictions across multiple perturbed inputs.

3 Random Soft Prompts and Why They Work

3.1 Random Soft Prompt

Random embedding vectors appended to LLM inputs, carrying no learned content, achieve accuracy comparable to optimized *soft prompt* methods across mathematical reasoning settings (Table 1, §4.2). This section lays the theoretical groundwork for why a training-free distribution suffices, by tracking the *attention mass* w placed on RSP tokens. Early in decoding w is large enough to open a different decoding branch (§3.2). As decoding proceeds, RSP attention empirically declines. Section 3.3 analyzes this behavior through the joint effects of KV cache growth and the evolving attention-logit gap. This constitutes an implicit explore-then-exploit process. Independent resampling provides a perturbation for each rollout, enabling exploration of different decoding trajectories (Appendix H).

To isolate the act of injection from learned content while keeping the magnitude comparable to real tokens and not privileging any direction in embedding space (Proposition 1), we draw RSP from an isotropic Gaussian fitted to the entrywise statistics of the pretrained embedding table. Alternative initializations score slightly lower on MATH-500 and AIME24, within seed noise (Appendix C). Let $\mathbf{W}_E \in \mathbb{R}^{V\times d}$ denote the pretrained token-embedding matrix, with V the vocabulary size and d the hidden dimension. Write its entrywise statistics as $\mu_E := \frac{1}{V\cdot d}\sum_{v,k}[\mathbf{W}_E]_{v,k}$ (the mean over all $V \cdot d$ entries) and $\sigma_E := \mathrm{std}(\mathbf{W}_E)$. An RSP of length L is a sequence of continuous vectors $\mathbf{H} = [h_1, \ldots, h_L] \in \mathbb{R}^{L\times d}$ drawn independently from

$$h_j \;\sim\; \mathcal{N}(\mu_E \mathbf{1}_d,\; \sigma_E^2 \mathbf{I}_d), \quad j = 1, \ldots, L. \tag{1}$$

The centered form $\bar{h}_j := h_j - \mu_E \mathbf{1}_d \sim \mathcal{N}(0, \sigma_E^2 \mathbf{I}_d)$ is a zero-mean isotropic Gaussian. Letting $\mathbf{X} \in \mathbb{R}^{T \times d}$ denote the input token-embedding sequence (the T rows of $\mathbf{W}_E$ corresponding to the prompt tokens), the main text uses the *suffix* form $[\mathbf{X}; \mathbf{H}] \in \mathbb{R}^{(T+L)\times d}$ as the default. Other positions (*prefix* $[\mathbf{H}; \mathbf{X}]$, *infix* ($\mathbf{H}$ inserted within $\mathbf{X}$) [Kang et al., 2026, Xu et al., 2025, Ye et al., 2026, Zhang et al., 2026]) are reported as ablations in §4.2 and Appendix A. RSP is not a learnable parameter, so it receives no gradient and is freshly drawn for each rollout. We define decoding step t to be a *branching event* and the last-layer hidden state $\mathbf{s}^{(t)}$ to be a *branching state* when the LM-head top-K differs from that of the no-RSP baseline $\bar{\mathbf{s}}^{(t)}$ (Figure 1, red).[1]

3.2 Exploration: RSP strengthens early-stage exploration

Inserting a random prompt concentrates attention on it, and that concentration amplifies exploration. The effect of a single injection on the hidden state raises two questions, *how much* and *where*, decided respectively by the scalar w inside one attention head and by the distribution of $\mathbf{H}$. The isotropic Gaussian addresses both at once. Both questions trace back to the branching condition of §3.1: for a single RSP draw to shift top-K, its perturbation of the head output must be large enough to propagate through the layers (*how much*) and along directions that affect logit ordering (*where*).

Consider one attention head in self-attention layer ℓ. At query position i, the query attends $n \geq 1$ unmasked real tokens (n depends on i via the causal mask) and $L \geq 1$ RSP tokens, all attention logits finite. Write the attention weights as $\alpha_{ij}^{(\ell)}$ (softmax-normalized so that $\sum_{j=1}^{n} \alpha_{ij}^{(\ell)} + \sum_{j=1}^{L} \alpha_{i,n+j}^{(\ell)} = 1$) and the value vectors on the real and RSP sides as $\mathbf{v}_j^{(\ell)}, \mathbf{v}_{r_j}^{(\ell)}$. Defining the attention mass on random tokens $w_{r,i}^{(\ell)} := \sum_{j=1}^{L} \alpha_{i,n+j}^{(\ell)}$, the real-token mass $1 - w_{r,i}^{(\ell)}$ is positive and the head output $\mathbf{o}_i^{(\ell)}$ splits *exactly* into a renormalized real-token term $\tilde{\mathbf{o}}_i^{(\ell)}$ and an RSP-induced contribution $\boldsymbol{\eta}_i^{(\ell)}$:

$$\mathbf{o}_i^{(\ell)} \;=\; (1 - w_{r,i}^{(\ell)})\, \tilde{\mathbf{o}}_i^{(\ell)} \;+\; \boldsymbol{\eta}_i^{(\ell)}, \quad \tilde{\mathbf{o}}_i^{(\ell)} := \sum_{j=1}^{n} \frac{\alpha_{ij}^{(\ell)}}{1 - w_{r,i}^{(\ell)}}\, \mathbf{v}_j^{(\ell)}, \quad \boldsymbol{\eta}_i^{(\ell)} := \sum_{j=1}^{L} \alpha_{i,n+j}^{(\ell)} \mathbf{v}_{r_j}^{(\ell)}. \tag{2}$$

Equation (2) follows exactly by factoring the positive real-token attention mass $1 - w_{r,i}^{(\ell)}$ out of the real-token contribution. This single quantity $w_{r,i}^{(\ell)}$ controls both the attenuation ratio $1 - w_{r,i}^{(\ell)}$ on the real signal and the upper bound on the random contribution $\|\boldsymbol{\eta}_i^{(\ell)}\| \leq w_{r,i}^{(\ell)} \max_j \|\mathbf{v}_{r_j}^{(\ell)}\|$.

The scalar w controls *how much*, while the distribution of $\mathbf{H}$ determines *where*. For the local logit argument, isolate one RSP position and let $\bar{h} \in \mathbb{R}^d$ denote the i.i.d. marginal of any centered vector $\bar{h}_j$ in Eq. (1), with the other centered RSP positions fixed at zero. Let $z_a(\bar{h})$ be the under-RSP output logit at vocab token a and write the vocab-logit gap $\Delta_{ab}(\bar{h}) := z_a(\bar{h}) - z_b(\bar{h})$ ($\bar{h} = 0$ is the centered mean, not the no-RSP state, as explained in Appendix G). The transformer's logit map is non-linear in $\bar{h}$, but a first-order Taylor expansion around $\bar{h} = 0$ gives the local surrogate $\Delta_{ab}(\bar{h}) \approx \Delta_{ab}(0) + b_{ab}^{\top} \bar{h}$, with $b_{ab} := \nabla_{\bar{h}}(z_a - z_b)|_{\bar{h}=0} \in \mathbb{R}^d$ the gradient direction along which vocab tokens a, b swap rank (not unit-norm in general). Because RSP is training-free, we cannot know in advance which b_{ab} opens a useful branch, so the distribution must avoid systematically under-perturbing any direction. A deterministic injection pins to one direction, while vocabulary sampling inherits the embedding table's anisotropy. At a fixed variance budget, the worst-case directional variance is maximized if and only if the covariance is isotropic.

Proposition 1 (Maximin directional coverage)**.** *Let $\mathcal{D}_\rho$ denote the family of zero-mean distributions on $\mathbb{R}^d$ whose covariance satisfies $\mathrm{tr}(\Sigma_D) \leq \rho^2$. For each $D \in \mathcal{D}_\rho$, $\min_{\|u\|=1} \mathrm{Var}_{h\sim D}(u^\top h) = \lambda_{\min}(\Sigma_D)$, and $\sup_{D \in \mathcal{D}_\rho} \lambda_{\min}(\Sigma_D) = \rho^2/d$, attained iff $\Sigma_D = (\rho^2/d)\mathbf{I}_d$.*

This is a *design criterion* for the prompt law, not a guarantee of correctness. Correctness enters through the task-side $p_{\min}(x)$ assumption of §3.3 and independent resampling.[2]

[1] Italic h_j denotes a vector in the RSP input space, and bold $\mathbf{s}^{(t)}$ denotes a hidden state. K denotes the cardinality in top-K analyses. Nucleus sampling uses a separate threshold p. t for decoding step, ℓ for self-attention layer.

[2] The maximin is stated in input space. Isotropy is not claimed to survive transformer layers. Any logit direction pulls back via the model Jacobian to an input-space b, and isotropy guarantees only that no such pulled-back direction (in the role of u above) has zero projected variance.

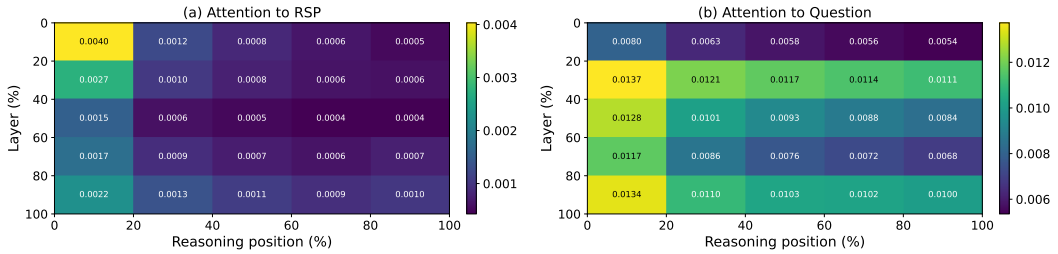


Figure 2: **Per-token attention mass during generation.** Results on Qwen2.5-Math-7B with *suffix* injection, averaged over 500 MATH-500 problems with independent RSP draws. (a) RSP-token attention decreases along the reasoning-position axis, consistent with the token-count and gap dynamics described by Theorem 1. (b) Question-token attention stays comparatively uniform. Both panels report attention mass per token, averaged over heads and samples, normalized by the RSP token count L in (a) and the question-token count in (b) (Appendix I).

Gaussian RSP attains the equality with $\Sigma = \sigma_E^2 \mathbf{I}_d$ ($\rho^2 = d\sigma_E^2$) and adds *full support* on $\mathbb{R}^d$ (Appendix G). Isotropy excludes no direction. Full support gives positive probability to the open branching-event region (§3.1) whenever it is non-empty (Appendix H.4.3). Monotone temperature preserves ranking and cannot reach this region. Together, these properties allow RSP to change token rankings and reach different branching states across independent draws early in reasoning.

3.3 Annealing: KV cache growth dilutes RSP attention

The early branching stabilizes as decoding proceeds. Each step adds one real token while the number of RSP tokens stays fixed. To characterize the resulting attention allocation, we express the RSP attention mass exactly in terms of token counts and a generation-dependent attention-logit gap.

Theorem 1 (RSP attention mass)**.** *Under the attention setup of §3.2, suppress the layer index ℓ and let d_{head} denote the per-head key dimension. Write the scaled logits as $s_{ij} := \mathbf{q}_i^\top \mathbf{k}_j / \sqrt{d_{head}}$ and $s_{ir_j} := \mathbf{q}_i^\top \mathbf{k}_{r_j} / \sqrt{d_{head}}$. The token-count-normalized log-sum-exp gap and the RSP attention mass are*

$$g_i := \log\Big(\tfrac{1}{n}\textstyle\sum_{j=1}^{n} e^{s_{ij}}\Big) - \log\Big(\tfrac{1}{L}\textstyle\sum_{j=1}^{L} e^{s_{ir_j}}\Big), \qquad w_{r,i} = \tfrac{L}{L+ne^{g_i}}. \tag{3}$$

Proof. Let $S_i := \sum_{j=1}^{n} e^{s_{ij}}$ and $R_i := \sum_{j=1}^{L} e^{s_{ir_j}}$. By definition, $e^{g_i} = (S_i/n)/(R_i/L)$, so softmax normalization gives $w_{r,i} = R_i/(S_i + R_i) = L/(L + ne^{g_i})$. □

Applied at each layer ℓ, Eq. (3) makes the generation dependence explicit: with L fixed, RSP attention decreases as $ne^{g_i^{(\ell)}}$ increases. Cache growth increases n, while the evolving queries and keys determine $g_i^{(\ell)}$. Under the same setting as Figure 2, the measured gap broadly increases along generation, most clearly in the shallowest and deepest layer bands (Appendix H.3). Together with the observed attention decay, this supports the annealing picture. Through Eq. (2), declining RSP attention reduces the total weight assigned to RSP value vectors, consistent with early exploration followed by stabilization.

Lifting single-shot branching to task accuracy across N rollouts requires (M1) a positive lower bound $p_{\min}(x) > 0$ on the per-rollout probability that execution reaches a correct trajectory and (M2) the N executions are independent. Then, $\Pr(\text{Pass@}N \text{ on } x) \geq 1-(1-p_{\min}(x))^N$ (Appendix H.4.4). §3.2's isotropy and full support make local branch opening reachable when the open region is nonempty, but they do not by themselves guarantee the task-side correctness condition (M1). *Independent resampling* supplies (M2). Section 4 evaluates the contribution of independent RSP resampling to Pass@N, comparing shared and independently resampled RSPs under temperature sampling.

4 Empirical Validation

We first evaluate whether untrained RSPs reach the performance range of optimized soft prompts (§4.2), then directly test the role of learned content through controlled interventions on the injected

Table 2: **Accuracy across model configurations and benchmarks.** RSP accuracy (%) is reported as mean ± std over five initialization seeds. The baseline uses greedy decoding.

	MATH-500			GSM8K			AIME24		
Model	Baseline	RSP(*prefix*)	RSP(*suffix*)	Baseline	RSP(*prefix*)	RSP(*suffix*)	Baseline	RSP(*prefix*)	RSP(*suffix*)
Instruct Models									
LLaMA-3.1-8B-Inst	52.20	41.24 ±2.16	49.52 ±1.48	85.75	76.30 ±0.75	86.70 ±0.39	6.67	3.33 ±2.36	6.67 ±3.33
Qwen2.5-Math-7B-Inst	83.20	81.72 ±0.46	83.52 ±0.56	95.45	94.92 ±0.28	95.60 ±0.08	13.33	13.33 ±2.36	16.67 ±0.00
Qwen2.5-Math-1.5B-Inst	73.00	75.60 ±0.51	74.12 ±1.36	85.29	85.26 ±0.39	84.99 ±0.22	10.00	12.00 ±1.83	13.33 ±5.27
Base Models + ChatML (Format Mismatch)									
Qwen2.5-Math-7B	52.20	57.40 ±1.71	69.40 ±1.03	58.30	56.25 ±1.14	75.88 ±0.36	23.33	9.33 ±4.35	20.00 ±2.36
Qwen2.5-Math-1.5B	34.40	61.56 ±1.42	48.80 ±2.09	37.45	66.07 ±1.03	50.24 ±2.09	13.33	14.67 ±2.98	8.00 ±5.06

vectors (§4.3). We next examine how injection reshapes the output distribution (§4.4) and whether these changes translate into trajectory diversity and Pass@N gains under independent resampling (§4.5). Finally, §4.6 evaluates whether RSP improves correct-solution collection and subsequent SFT.

4.1 Experimental Setup

We evaluate on three mathematical reasoning benchmarks, MATH-500 [Lightman et al., 2024], GSM8K [Cobbe et al., 2021], and AIME24, using LLaMA-3.1-8B-Instruct [Grattafiori et al., 2024] and the Qwen2.5-Math model family [Yang et al., 2024] across instruct models and base models with mismatched chat templates. We additionally evaluate the general-purpose Qwen2.5-{1.5B, 7B}-Instruct models on code generation (HumanEval+, MBPP+) [Liu et al., 2023], moral reasoning (ETHICS-commonsense, ETHICS-justice) [Hendrycks et al., 2021a], general language understanding (MMLU) [Hendrycks et al., 2021b], and commonsense reasoning (PIQA) [Bisk et al., 2020]. Tables 2 and 1 use greedy decoding to isolate the effect of RSPs from sampling stochasticity. The diversity and Pass@N experiments in §4.5 use the sampling settings specified there. RSPs follow the definition of §3.1, using the default *suffix* position with a drawn $\mathbf{H}$ per rollout. Since Theorem 1 expresses RSP attention mass in terms of the RSP length L, real-token count, and attention-logit gap, we compare $L \in \{10, 15, 20\}$ for each model and benchmark to identify an appropriate perturbation strength. Tables 2 and 1 report results using the selected lengths. Table 2 reports mean accuracy and standard deviation over five RSP initialization seeds. For TTSV, SoftCoT, and LTPO, we repeat training or optimization with five initialization seeds while keeping the hyperparameters in Appendix A.3 fixed. The mechanism analyses in §4.4–§4.5 use a fixed setting (L=20, *suffix*), which insulates them from this selection effect. Full experimental details and per-position results are in Appendix A.

4.2 How Does Untrained RSP Compare to Baselines & Optimized Soft Prompts?

Mathematical reasoning. Compared with three optimized methods (TTSV, SoftCoT, LTPO) under a unified evaluation pipeline (Appendix A.2), RSP reaches a similar accuracy range without learning the injected vectors (Table 1, presented in §3). What we want to highlight is the converse: a fixed, training-free distribution reaching this range provides evidence that part of the *soft-prompt* effect comes less from learned content than from the random directional perturbation that injection induces.

Beyond the direct comparison with optimized methods, we further evaluate RSP across two injection positions (*prefix*, *suffix*) and several model configurations, unpacking the per-configuration patterns of Table 2 with five RSP initialization seeds. Where the baseline is already strong, *suffix* RSP achieves comparable accuracy on the Qwen2.5-Math-Instruct models. On the harder AIME24 benchmark, *suffix* RSP improves mean accuracy by 3.3 pp at both model scales. Another striking pattern is format-mismatch recovery: when a ChatML template is applied to a base model not trained for that format, *suffix* recovers $+17.20/+17.58$ pp on Qwen2.5-Math-7B (MATH-500/GSM8K), and *prefix* recovers $+27.16/+28.62$ pp on Qwen2.5-Math-1.5B.

Evaluation on additional benchmarks. Prior soft-prompt work primarily reports results on mathematical reasoning. We extend the evaluation to six additional benchmarks across four domains: code, moral reasoning, general language understanding, and commonsense. We use the general-purpose Qwen2.5-{1.5B, 7B}-Instruct models and average results over five seeds.

Table 3: **Accuracy beyond mathematics.** Results on Qwen2.5-{1.5B, 7B}-Instruct. RSP accuracy (%) is reported as mean ± std over five initialization seeds. ETHICS-C and ETHICS-J denote ETHICS-commonsense and ETHICS-justice.

Model	Method	HumanEval+	MBPP+	ETHICS-C	ETHICS-J	MMLU	PIQA
Qwen2.5-1.5B	Baseline	45.70	48.10	60.82	60.36	61.34	77.15
	RSP(*prefix*)	51.92 ± 1.61	51.60 ± 0.54	63.74 ± 0.37	59.68 ± 0.23	61.13 ± 0.13	77.17 ± 0.25
	RSP(*suffix*)	51.60 ± 1.98	52.42 ± 0.88	59.63 ± 0.27	60.60 ± 0.11	61.22 ± 0.23	76.25 ± 0.20
Qwen2.5-7B	Baseline	71.30	65.60	76.65	76.59	73.14	87.43
	RSP(*prefix*)	73.80 ± 1.20	67.70 ± 0.27	76.81 ± 0.27	74.96 ± 0.39	72.90 ± 0.12	86.59 ± 0.22
	RSP(*suffix*)	74.16 ± 1.00	69.86 ± 0.47	76.64 ± 0.08	76.91 ± 0.12	73.29 ± 0.09	87.59 ± 0.10

Table 4: **Intervening on learned prompt content.** Results on Qwen2.5-Math-1.5B-Instruct. Accuracy (%) is reported as mean ± std over three seeds. Injection length and position remain fixed within each method. **Bold** denotes the highest mean accuracy in each column.

	TTSV			LTPO			SoftCoT		
Intervention	MATH-500	GSM8K	AIME24	MATH-500	GSM8K	AIME24	MATH-500	GSM8K	AIME24
Trained	72.33 ±1.46	84.08 ±1.08	6.67 ±0.00	74.07 ±0.82	84.69 ±0.21	7.78 ±1.57	72.87 ±1.18	84.84 ±1.24	8.89 ±1.57
50% front	**75.40** ±0.28	84.99 ±0.70	10.00 ±0.00	**74.27** ±0.47	84.63 ±0.34	10.00 ±2.72	72.47 ±0.98	84.15 ±1.29	10.00 ±2.72
50% random	72.93 ±1.95	81.07 ±3.54	**12.22** ±1.57	73.87 ±0.90	84.91 ±0.39	8.89 ±1.57	73.33 ±0.25	84.53 ±0.32	12.22 ±1.57
100% RSP	75.13 ±0.25	**85.09** ±0.44	11.11 ±1.57	73.80 ±0.91	**85.06** ±0.57	11.11 ±1.57	73.67 ±0.52	**85.01** ±0.19	8.89 ±1.57
Token-shuffle	72.40 ±0.99	83.83 ±0.38	6.67 ±2.72	73.60 ±0.49	84.61 ±0.06	7.78 ±1.57	72.80 ±1.56	84.96 ±0.29	11.11 ±1.57
Dim-permute	71.93 ±0.57	83.62 ±0.55	5.56 ±1.57	74.00 ±0.43	84.00 ±0.77	**12.22** ±1.57	**74.80** ±0.16	84.99 ±0.16	**13.33** ±0.00

Table 3 shows consistent gains on code (+2.1 to +6.2 pp on HumanEval+/MBPP+), with the 1.5B model benefiting most. On short-answer MCQ tasks (ETHICS, MMLU, and PIQA), RSP performs comparably to the baseline. Together with the AIME24 results, these gains are consistent with the mechanism described in Section 3: RSP can perturb early decoding decisions and, across independent draws, enable exploration of different reasoning trajectories. This may benefit tasks involving extended reasoning and generation, such as hard mathematics and code. Section 4.5 directly examines whether this trajectory diversity translates into Pass@N gains.

4.3 Does Performance Depend on Learned Prompt Content?

Building on the results in §4.2, we examine the contributions of learned prompt content and embedding injection in greater detail. Following the settings in the original papers, we keep the injection token count and position fixed within each method while varying the injected vectors: 20 *prefix* tokens for TTSV, 4 *infix* tokens for SoftCoT, and 8 *infix* tokens for LTPO. We evaluate three complementary probes (i)–(iii) on Qwen2.5-Math-1.5B-Instruct across MATH-500, GSM8K, and AIME24 (Tables 4 and 5). All results report mean accuracy and standard deviation over three seeds (seed details in Appendix A.3). Appendix B reports additional results on Qwen2.5-Math-7B-Instruct.

(i) Is learned content necessary to retain performance? We replace the first half, a randomly selected half, or all trained vectors with independent draws from the RSP distribution in Eq. (1). Removing trained vectors does not produce a consistent loss of accuracy. On the 1.5B model, full replacement raises TTSV's MATH-500 accuracy from 72.33% to 75.13%, while LTPO changes from 74.07% to 73.80% and SoftCoT from 72.87% to 73.67%. Partial replacement depends on which positions are replaced, as illustrated by TTSV on GSM8K, but retaining more trained vectors does not consistently yield higher accuracy.

(ii) Do trained vectors encode useful information? Since random replacement does not consistently reduce accuracy, we next ask whether the trained vectors actually encode useful information. *Token-shuffle* disrupts their order, while *Dim-permute* independently permutes the coordinates within each vector. These interventions disrupt the organization of the learned content while preserving vector norms. Neither causes a consistent accuracy drop across methods and benchmarks. On MATH-500, dimension permutation changes TTSV's accuracy from 72.33% to 71.93% and LTPO's from 74.07% to 74.00%, while increasing SoftCoT's accuracy from 72.87% to 74.80%.

(iii) Is the model simply insensitive to injected vectors? Disrupting the learned content also leaves accuracy largely unchanged. This raises the possibility that the model simply ignores the injected vectors. To test this, we increase the learning rate by $1000\times$ when training TTSV and SoftCoT, keeping the injection token count and position fixed. Table 5 shows sharp accuracy drops on MATH-500 and GSM8K. TTSV falls to 4.20% and 2.60%, and SoftCoT to 30.60% and 36.01%, respectively. Taken together, probes (i)–(iii) suggest that, in the evaluated settings, the observed gains arise primarily from embedding injection itself rather than the learned content of the injected vectors.

Table 5: **Sensitivity to overtrained embeddings.** Results on Qwen2.5-Math-1.5B-Instruct. Accuracy (%) is reported as mean ± std over three seeds. Increasing the learning rate by $1000\times$ sharply reduces MATH-500 and GSM8K accuracy.

	TTSV		SoftCoT	
Benchmark	Trained	Overtrained	Trained	Overtrained
MATH-500	72.33 ±1.46	4.20 ±2.27	72.87 ±1.18	30.60 ±22.41
GSM8K	84.08 ±1.08	2.60 ±3.41	84.84 ±1.24	36.01 ±25.08
AIME24	6.67 ±0.00	8.89 ±4.16	8.89 ±1.57	3.33 ±4.71

4.4 How Does RSP Affect the Output Distribution?

The results in Section 4.3 motivate examining whether RSP and optimized soft prompts share common patterns in their output distributions despite differences in learned content and training objectives. We compare their generation dynamics using Qwen2.5-Math-1.5B-Instruct on MATH-500, measuring per-token entropy, top-1 probability, and varentropy across generation steps. Figure 3 compares these metrics over the first 5% of generation steps for RSP and prior *soft prompt* methods (LTPO, SoftCoT, TTSV). Full curves are provided in Appendix A.5.

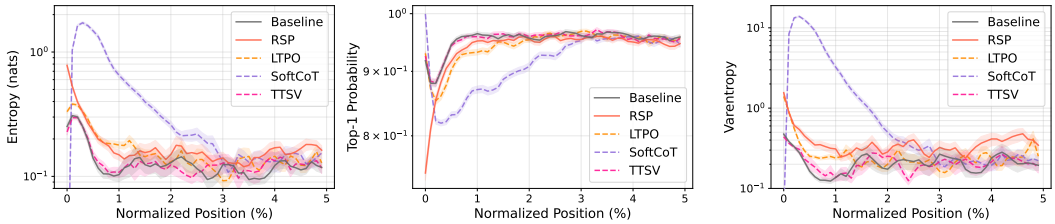


Figure 3: Mean entropy, top-1 probability, and varentropy during the first 5% of generation steps (Qwen2.5-Math-1.5B-Instruct, MATH-500). Shading indicates ±1 SEM. Solid lines: RSP and the baseline. Dashed lines: prior *soft prompt* methods (LTPO, SoftCoT, TTSV).

RSP and optimized soft prompt methods exhibit similar temporal patterns despite their different training objectives. RSP shows early-stage entropy↑, top-1 probability↓, and varentropy↑. When continuous embeddings outside the vocabulary enter the input, attention to the additional keys can redistribute probability mass across multiple high-probability candidates in the early next-token distribution. The joint pattern is consistent with this redistribution, a precondition for the trajectory branching of §3.2. The optimized methods show related early behavior. Even TTSV, which explicitly minimizes mean trajectory entropy, exhibits early entropy comparable to the baseline. Later in generation, the metrics converge toward the baseline, consistent with the attenuation analyzed in §3.3.

4.5 Does RSP Induce Trajectory Diversity?

Section 4.4 showed that RSP reshapes the early-stage output distribution. We now examine whether this shift is accompanied by greater diversity in token rankings, the semantic content of reasoning trajectories, and task-level outcomes. These analyses connect changes in local token distributions with diversity across complete trajectories and gains in Pass@N. We use Qwen2.5-Math-1.5B-Instruct on MATH-500 with *suffix* injection and $L = 20$, and additionally evaluate Pass@N on AIME24.

Token-level: distribution beyond temperature. We quantify how much RSP shifts the first-token distribution, the critical forking point for reasoning trajectories [Wang et al., 2025]. Using the baseline at τ=1.0 as reference, we compare baselines at τ=2.0, 3.0 (16 samples per problem) against RSP at τ=1.0 (averaged over 16 seeds) on three metrics: Spearman rank correlation ρ to detect rank reorganization, the probability mass placed outside the baseline's top-10 to measure support expansion, and Jensen–Shannon divergence to quantify the overall magnitude of distributional change.

Table 6 shows the contrast: RSP's distributional shift falls between τ=2.0 and τ=3.0 in magnitude, but the mechanism differs. Temperature is a monotone transform that preserves ranking ($\rho = 1$ at any τ), while RSP partially preserves but reranks ($\rho = 0.709$). The accuracy row exposes the cost: matching RSP's magnitude with temperature toward τ=3.0 collapses accuracy to 1.42%, whereas RSP preserves 73.30%. Thus, RSP produces a fundamentally different perturbation than temperature scaling. Moreover, RSP's accuracy variability across 16 seeds at τ=1.0 is $\pm$1.07, comparable to the $\pm$1.47 that temperature sampling produces across 16 samples at τ=2.0 (details in Appendix D).

Table 6: First-token distribution metrics measured against the baseline at τ=1.0, comparing baselines at higher temperatures (τ=2.0, 3.0) with RSP at τ=1.0. RSP metrics are averaged over 16 random seeds, and Acc reports mean $\pm$ std across 16 samples per problem.

Metric	τ=2.0	τ=3.0	**RSP**
Spearman ρ	1.000	1.000	**0.709**
Mass outside top-10	5.18%	29.11%	**21.86%**
JS divergence	0.060	0.218	**0.131**
Acc (%)	20.77 $\pm$ 1.47	1.42 $\pm$ 0.35	**73.30 $\pm$ 1.07**

Trajectory-level: semantic diversity. Token-level reranking raises a second question: do first-token perturbations diverge into distinct trajectories, or converge to similar ones? We sample 64 problems from MATH-500 and generate 300 independent trajectories per problem under Baseline and RSP at $\tau = 1.0$. Each trajectory is encoded by BGE-M3 [Chen et al., 2024] into a dense semantic vector, and we compute three metrics: *pairwise cosine distance* (overall semantic spread), *inter-cluster distance* between centroids of DBSCAN [Ester et al., 1996], a density-based clustering algorithm (separation between distinct trajectory groups), and *intra-cluster distance* (coherence within groups). Table 7 reveals three concurrent patterns: pairwise distance rises by 1.4×, inter-cluster distance jumps by 5.4×, and intra-cluster distance is essentially unchanged. The trajectory set becomes more diverse and splits into more separated groups while preserving within-group coherence. RSP also yields larger pairwise distance than baseline on 56 of 64 problems.

Table 7: Semantic diversity metrics (64 problems, 300 trajectories per condition).

Metric	**Baseline**	**RSP**
Pairwise cos. dist.	0.0612	**0.0879**
Inter-cluster dist.	0.0183	**0.0982**
Intra-cluster dist.	0.0526	0.0528

Outcome-level: Pass@N scaling. Does this token- and trajectory-level diversity translate into task-level outcome diversity, measured by Pass@N [Chen et al., 2021] (the probability that at least one of N sampled solutions is correct)? We generate 16 samples per problem on MATH-500 and AIME24 and compare four conditions. (i) *Baseline* uses $\tau = 1.0$. (ii) *RSP (single seed)* uses $\tau = 1.0$ with one RSP shared across all samples. (iii) *RSP (indep. seed)* uses greedy decoding with a fresh RSP per sample. (iv) *RSP (indep. seed)* uses $\tau = 1.0$ with a fresh RSP per sample.

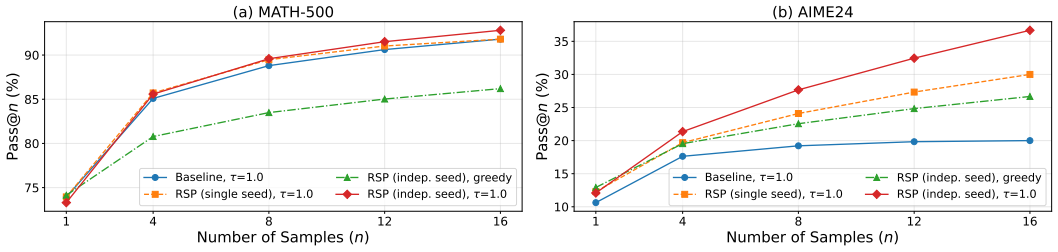


Figure 4: Pass@N scaling on (a) MATH-500 and (b) AIME24 with Qwen2.5-Math-1.5B-Instruct, 16 samples per problem. *Baseline*: temperature sampling only. *RSP (single seed)*: single RSP shared across samples combined with temperature. *RSP (indep. seed)*: a different RSP per sample, with or without temperature.

Condition *(iv)* achieves the highest Pass@N on both benchmarks, reaching 92.80% on MATH-500 and 36.67% on AIME24 at $N = 16$ (intermediate and annealed temperatures do not close the gap, Appendix E). With a shared RSP, condition *(ii)* matches the baseline on MATH-500 and improves AIME24 from 20.00% to 30.00%, but remains below independent resampling on both benchmarks. Pass@1 stays comparable to the baseline on MATH-500, and on the harder AIME24 *(iv)* even surpasses it, suggesting that independent RSPs combined with temperature sampling expand trajectory diversity without sacrificing single-sample accuracy. Qualitative analysis is in Appendix F.

4.6 Application: Rejection-Sampling SFT with RSP

Section 4.5 showed that independent RSP resampling expands trajectory diversity and improves Pass@N, including on the challenging AIME24 benchmark. We now ask whether this advantage can improve the collection of correct solutions for supervised fine-tuning (SFT), focusing on problems for which the initial model has a low solution success rate. We study rejection-sampling SFT [Singh et al., 2024] using Qwen2.5-Math-7B-Instruct. We define a hard subset of OpenR1-Math-46k [Yan et al., 2025] as the 32,737 problems that this model solves at most once in four rollouts without RSP. For each hard problem, Baseline and RSP each generate eight rollouts at $\tau = 1.0$. We retain only correct solutions and fine-tune the same initial model separately on each collected dataset for one epoch. We evaluate the model without SFT and the two models fine-tuned on solutions collected by Baseline and RSP on Minerva [Lewkowycz et al., 2022], GaoKao [Liao et al., 2024], OlympiadBench [He et al., 2024], AIME24, and AMC23. All evaluations are performed without RSP. Results are averaged over three data-collection seeds (training and seed details in Appendix A.4).

Table 8: **Correct-solution collection on hard problems.** Results on the fixed hard subset using Qwen2.5-Math-7B-Instruct, with eight rollouts per problem. Percentages are reported as mean ± std over three collection seeds. Coverage is the fraction of problems with at least one correct rollout.

Method	Problem coverage (%)	Rollout accuracy (%)
Baseline	23.75 ± 0.10	5.54 ± 0.03
RSP	49.68 ± 0.13	17.45 ± 0.05

Table 8 shows that RSP collects at least one correct solution for 49.68% of the hard problems (problem coverage), compared with 23.75% for Baseline, approximately $2.1\times$ the coverage under the same rollout budget. Per-rollout accuracy also increases from 5.54% to 17.45%. Thus, RSP provides a larger pool of correct solutions spanning more hard problems for subsequent SFT.

Table 9: **SFT with self-collected correct solutions.** Evaluation without RSP on Qwen2.5-Math-7B-Instruct. Accuracy (%) is reported as mean ± std over three collection seeds for the SFT conditions. AIME24 and AMC23 use Avg@32. Avg. denotes the mean across five benchmarks.

Method	Minerva	GaoKao	Olympiad	AIME24	AMC23	Avg
No SFT	39.7	66.8	40.4	6.7	47.7	40.26
Baseline-SFT	42.7 ± 1.6	67.0 ± 0.5	39.4 ± 1.1	9.2 ± 0.6	52.2 ± 1.0	42.11 ± 0.54
RSP-SFT	43.2 ± 0.8	69.4 ± 1.2	40.7 ± 0.8	10.0 ± 0.1	55.5 ± 1.8	43.77 ± 0.54

The collection advantage also transfers to subsequent training (Table 9). RSP-SFT reaches a five-benchmark average of 43.77%, compared with 42.11% for Baseline-SFT and 40.26% without SFT. The 1.66 pp gain over Baseline-SFT suggests that RSP can improve training-data collection as well as inference by obtaining correct solutions for more hard problems. This comparison includes differences in both the amount and composition of the collected data. Additional DAPO results are reported in Appendix J.

5 Conclusion

We studied Random Soft Prompts (RSPs) to distinguish the contribution of learned content from the effect of embedding injection itself. Across the evaluated settings, untrained random vectors reach the performance range of optimized *soft prompts*, while replacing or disrupting trained vectors does not consistently reduce accuracy. These results suggest that embedding injection itself contributes to the gains observed with trained *soft prompts*.

Our analysis connects RSP-induced trajectory diversity to perturbations of early decoding decisions and the subsequent attenuation of RSP attention. We express this attention mass in terms of token counts and the evolving attention-logit gap. Empirically, RSP exhibits early output-distribution patterns similar to optimized *soft prompts*, increases trajectory diversity, and improves Pass@N under independent resampling. Beyond inference, RSP collects correct solutions for more hard problems and improves subsequent SFT. Together, these findings identify embedding injection as a source of reasoning diversity that can benefit both inference and training-data collection.

Acknowledgements

We thank Laura Mascarell, Woochang Sim, Jumyung Park, and the anonymous reviewers for their valuable feedback. This work was supported by IITP (RS-2024-00436680, 40%; RS-2024-00445087, 20%; RS-2023-00216011, 10%; RS-2026-25614465, 10%; No. 2019-0-01842; 10%), NRF (RS-2024-00345809, 10%) funded by the Ministry of Science and ICT, Korea. Computing resource is supported by AICA, GIST SCENT-AI, and Advanced GPU Utilization Support Program (No. 02-26-01-0181).

References

Nora Belrose, Igor Ostrovsky, Lev McKinney, Zach Furman, Logan Smith, Danny Halawi, Stella Biderman, and Jacob Steinhardt. Eliciting Latent Predictions from Transformers with the Tuned Lens. *arXiv:2303.08112*, 2023.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about Physical Commonsense in Natural Language. In *AAAI*, 2020.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. M3-Embedding: Multi-Linguality, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. In *Findings of ACL*, 2024.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating Large Language Models Trained on Code. *arXiv:2107.03374*, 2021.

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:2110.14168*, 2021.

Jeremy M. Cohen, Elan Rosenfeld, and J. Zico Kolter. Certified Adversarial Robustness via Randomized Smoothing. In *ICML*, 2019.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *KDD*, 1996.

Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Ian Osband, Alex Graves, Vlad Mnih, Remi Munos, Demis Hassabis, Olivier Pietquin, Charles Blundell, and Shane Legg. Noisy Networks for Exploration. In *ICLR*, 2018.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer Feed-Forward Layers Are Key-Value Memories. In *EMNLP*, 2021.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. Think before You Speak: Training Language Models with Pause Tokens. In *ICLR*, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song,

Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca

Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The Llama 3 Herd of Models. *arXiv:2407.21783*, 2024.

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. In *ICLR*, 2025.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A Challenging Benchmark for Promoting AGI with Olympiad-Level Bilingual Multimodal Scientific Problems. In *ACL*, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. Aligning AI With Shared Human Values. In *ICLR*, 2021a.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring Massive Multitask Language Understanding. In *ICLR*, 2021b.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-Efficient Transfer Learning for NLP. In *ICML*, 2019.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. In *ICLR*, 2022.

Neel Jain, Ping-yeh Chiang, Yuxin Wen, John Kirchenbauer, Hong-Min Chu, Gowthami Somepalli, Brian R. Bartoldson, Bhavya Kailkhura, Avi Schwarzschild, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. NEFTune: Noisy Embeddings Improve Instruction Finetuning. In *ICLR*, 2024.

Xinyue Kang, Diwei Shi, and Li Chen. Model Whisper: Steering Vectors Unlock Large Language Models' Potential in Test-time. In *AAAI*, 2026.

Brian Lester, Rami Al-Rfou, and Noah Constant. The Power of Scale for Parameter-Efficient Prompt Tuning. In *EMNLP*, 2021.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving Quantitative Reasoning Problems with Language Models. In *NeurIPS*, 2022.

Xiang Lisa Li and Percy Liang. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *ACL-IJCNLP*, 2021.

Minpeng Liao, Chengxi Li, Wei Luo, Jing Wu, and Kai Fan. MARIO: MAth Reasoning with code Interpreter Output - A Reproducible Pipeline. In *Findings of ACL*, 2024.

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. In *ICLR*, 2024.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is Your Code Generated by ChatGPT Really Correct? Rigorous Evaluation of Large Language Models for Code Generation. In *NeurIPS*, 2023.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. GPT Understands, Too. *AI Open*, 5:208–215, 2024.

Aman Madaan and Amir Yazdanbakhsh. Text and Patterns: For Effective Chain of Thought, It Takes Two to Tango. *arXiv:2209.07686*, 2022.

Aleksandar Petrov, Philip H. S. Torr, and Adel Bibi. When Do Prompting and Prefix-Tuning Work? A Theory of Capabilities and Limitations. In *ICLR*, 2024.

Alexander Robey, Eric Wong, Hamed Hassani, and George J. Pappas. SmoothLLM: Defending Large Language Models Against Jailbreaking Attacks. *Transactions on Machine Learning Research*, 2025.

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. HybridFlow: A Flexible and Efficient RLHF Framework. In *EuroSys*, 2025.

Avi Singh, John D. Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J. Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron Parisi, Abhishek Kumar, Alex Alemi, Alex Rizkowsky, Azade Nova, Ben Adlam, Bernd Bohnet, Gamaleldin Elsayed, Hanie Sedghi, Igor Mordatch, Isabelle Simpson, Izzeddin Gur, Jasper Snoek, Jeffrey Pennington, Jiri Hron, Kathleen Kenealy, Kevin Swersky, Kshiteej Mahajan, Laura Culp, Lechao Xiao, Maxwell L. Bileschi, Noah Constant, Roman Novak, Rosanne Liu, Tris Warkentin, Yundi Qian, Yamini Bansal, Ethan Dyer, Behnam Neyshabur, Jascha Sohl-Dickstein, and Noah Fiedel. Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models. *Transactions on Machine Learning Research*, 2024.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 Rule: High-Entropy Minority Tokens Drive Effective Reinforcement Learning for LLM Reasoning. In *NeurIPS*, 2025.

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. SoftCoT: Soft Chain-of-Thought for Efficient Reasoning with LLMs. In *ACL*, 2025.

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. Learning to Reason under Off-Policy Guidance. In *NeurIPS*, 2025.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-Math Technical Report: Toward Mathematical Expert Model via Self-Improvement. *arXiv:2409.12122*, 2024.

Wengao Ye, Yan Liang, and Lianlei Shan. Thinking on the Fly: Test-Time Reasoning Enhancement via Latent Thought Policy Optimization. In *ICLR*, 2026.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: An Open-Source LLM Reinforcement Learning System at Scale. In *NeurIPS*, 2025.

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

Guibin Zhang, Muxin Fu, and Shuicheng Yan. MemGen: Weaving Generative Latent Memory for Self-Evolving Agents. In *ICLR*, 2026.

A Experimental Setup and Full Accuracy Breakdown

The math accuracy experiments in Tables 1 and 2 are conducted on a single NVIDIA RTX A6000 48GB GPU with greedy decoding, a maximum generation length of 3,072 tokens for MATH-500 and GSM8K, and 4,096 tokens for AIME24. Batch size is fixed per model (16 for LLaMA-3.1-8B-Instruct and Qwen2.5-Math-7B variants, 32 for Qwen2.5-Math-1.5B variants). The RSP token count L is selected per (model, benchmark) pair from $\{10, 15, 20\}$ (Table 11). The single-seed results across *prefix*, *infix*, and *suffix* injection positions are reported in Table 10.

Table 10: Accuracy (%) across model configurations, injection positions, and benchmarks. Values in parentheses indicate the difference from the baseline. **Bold** denotes the best result and underline denotes the second best for each model–benchmark pair.

Model	Mode	MATH-500	GSM8K	AIME24
Instruct Models				
LLaMA-3.1-8B-Instruct	Baseline	**52.20**	85.75	6.67
	Prefix	45.00 (−7.2)	76.35 (−9.4)	3.33 (−3.3)
	Infix	49.40 (−2.8)	85.97 (+0.2)	**10.00** (+3.3)
	Suffix	49.40 (−2.8)	**86.73** (+1.0)	**10.00** (+3.3)
Qwen2.5-Math-7B-Instruct	Baseline	83.20	95.45	13.33
	Prefix	81.40 (−1.8)	94.92 (−0.5)	13.33 (+0.0)
	Infix	**84.20** (+1.0)	95.60 (+0.2)	**16.67** (+3.3)
	Suffix	83.40 (+0.2)	**95.68** (+0.2)	**16.67** (+3.3)
Qwen2.5-Math-1.5B-Instruct	Baseline	73.00	85.29	10.00
	Prefix	74.80 (+1.8)	85.29 (+0.0)	13.33 (+3.3)
	Infix	**75.20** (+2.2)	**85.75** (+0.5)	6.67 (−3.3)
	Suffix	74.20 (+1.2)	84.69 (−0.6)	**20.00** (+10.0)
Base Models (Raw Text)				
Qwen2.5-Math-7B	Baseline	**72.20**	84.15	6.67
	Prefix	71.60 (−0.6)	**84.31** (+0.2)	**16.67** (+10.0)
	Infix	63.20 (−9.0)	64.29 (−19.9)	**16.67** (+10.0)
	Suffix	55.60 (−16.6)	54.13 (−30.0)	13.33 (+6.7)
Qwen2.5-Math-1.5B	Baseline	61.40	**77.86**	**16.67**
	Prefix	**64.40** (+3.0)	74.30 (−3.6)	10.00 (−6.7)
	Infix	63.20 (+1.8)	73.92 (−3.9)	10.00 (−6.7)
	Suffix	54.60 (−6.8)	58.61 (−19.3)	3.33 (−13.3)
Base Models + ChatML (Format Mismatch)				
Qwen2.5-Math-7B	Baseline	52.20	58.30	**23.33**
	Prefix	59.20 (+7.0)	56.41 (−1.9)	3.33 (−20.0)
	Infix	62.00 (+9.8)	69.07 (+10.8)	**23.33** (+0.0)
	Suffix	**70.40** (+18.2)	**75.97** (+17.7)	**23.33** (+0.0)
Qwen2.5-Math-1.5B	Baseline	34.40	37.45	13.33
	Prefix	**63.40** (+29.0)	**67.02** (+29.6)	**20.00** (+6.7)
	Infix	39.20 (+4.8)	50.42 (+13.0)	6.67 (−6.7)
	Suffix	51.00 (+16.6)	50.64 (+13.2)	13.33 (+0.0)

Table 11: Number of RSP tokens (L) used for each model and benchmark.

Model	MATH-500	GSM8K	AIME24
LLaMA-3.1-8B-Instruct	15	20	15
Qwen2.5-Math-7B-Instruct	20	20	20
Qwen2.5-Math-1.5B-Instruct	20	10	20
Qwen2.5-Math-7B	10	10	20
Qwen2.5-Math-1.5B	10	10	20
Qwen2.5-Math-1.5B (ChatML)	20	20	10
Qwen2.5-Math-7B (ChatML)	20	20	20

A.1 RSP Injection Positions and Prompt Templates

Below we show the prompt structure for each injection position across all model types. Blue text indicates RSP embeddings, and purple text indicates special tokens.

A.1.1 *Prefix*

RSP embeddings are concatenated before the prompt embeddings. No text modification is applied.

ChatML (Qwen Instruct / Base + ChatML).

```
[Random Embeddings (L tokens)]
<|im_start|>system
Please reason step by step, and put your final answer within \boxed{}.<|im_end|>
<|im_start|>user
{question}<|im_end|>
<|im_start|>assistant
```

LLaMA Chat Template.

```
[Random Embeddings (L tokens)]
<|begin_of_text|>
<|start_header_id|>system<|end_header_id|>
Cutting Knowledge Date: December 2023
Today Date: 26 Jul 2024
Please reason step by step, and put your final answer within \boxed{}.<|eot_id|>
<|start_header_id|>user<|end_header_id|>
{question}<|eot_id|>
<|start_header_id|>assistant<|end_header_id|>
```

Raw Text (Qwen Base).

```
[Random Embeddings (L tokens)]
{question}
Please reason step by step, and put your final answer within \boxed{}.
```

A.1.2 *Infix*

A description text and special tokens are inserted into the prompt. The special token positions are then replaced with RSP embeddings at the embedding level.

ChatML (Qwen Instruct / Base + ChatML).

```
<|im_start|>system
Please reason step by step, and put your final answer within \boxed{}.<|im_end|>
<|im_start|>user
{question}
There are {L} special tokens that contain compressed latent reasoning information that
might be useful for your reasoning. If these tokens are useful for your case, you can
use them as reference. If these tokens are not useful for your case, you can ignore
them and focus back to solving the problem.
Here are the {L} special tokens: <special_token_1>...<special_token_L>
<|im_end|>
<|im_start|>assistant
```

LLaMA Chat Template.

```
<|begin_of_text|>
<|start_header_id|>system<|end_header_id|>
Cutting Knowledge Date: December 2023
Today Date: 26 Jul 2024
Please reason step by step, and put your final answer within \boxed{}.<|eot_id|>
<|start_header_id|>user<|end_header_id|>
{question}
There are {L} special tokens that contain compressed latent reasoning information that
might be useful for your reasoning. If these tokens are useful for your case, you can
use them as reference. If these tokens are not useful for your case, you can ignore
them and focus back to solving the problem.
Here are the {L} special tokens:
<reserved_special_token_0>...
<reserved_special_token_L>
```

```
<|eot_id|>
<|start_header_id|>assistant<|end_header_id|>
```

Raw Text (Qwen Base).

```
{question}
There are {L} special tokens that contain compressed latent reasoning information that
might be useful for your reasoning. If these tokens are useful for your case, you can
use them as reference. If these tokens are not useful for your case, you can ignore
them and focus back to solving the problem.
Here are the {L} special tokens: <special_token_1>...<special_token_L>
Please reason step by step, and put your final answer within \boxed{}.
```

A.1.3 *Suffix*

For chat-templated models, RSP embeddings are inserted immediately before the end-of-turn token. For raw text models, RSP embeddings are concatenated at the end of the prompt.

ChatML (Qwen Instruct / Base + ChatML).

```
<|im_start|>system
Please reason step by step, and put your final answer within \boxed{}.<|im_end|>
<|im_start|>user
{question}[Random Embeddings (L tokens)]<|im_end|>
<|im_start|>assistant
```

LLaMA Chat Template.

```
<|begin_of_text|>
<|start_header_id|>system<|end_header_id|>
Cutting Knowledge Date: December 2023
Today Date: 26 Jul 2024
Please reason step by step, and put your final answer within \boxed{}.<|eot_id|>
<|start_header_id|>user<|end_header_id|>
{question}[Random Embeddings (L tokens)]<|eot_id|>
<|start_header_id|>assistant<|end_header_id|>
```

Raw Text (Qwen Base).

```
{question}
Please reason step by step, and put your final answer within \boxed{}.[Random
Embeddings (L tokens)]
```

A.2 Answer Extraction and Grading

The final answer is extracted from the model output through a fallback chain. Each step is attempted in order. If extraction fails, the next step is tried.

Answer Extraction Fallback Chain

1. `"final answer is $...$. I hope"` pattern (Minerva format)
2. `\boxed{...}`: last non-empty box is used, and empty `\boxed{}` are skipped
3. Text following `"the answer is"`
4. Text following `"final answer is"`
5. Last number in the output (regex fallback)

Extracted answers are normalized by removing `$`, `\left`, `\right`, and unit strings. Grading is performed via symbolic equivalence checking: exact string match, numerical comparison (`math.isclose`, `rel_tol=1e-4`), and SymPy symbolic simplification.

A.3 Reproduced Prior Work Hyperparameters

All prior methods are reproduced on Qwen2.5-Math-1.5B-Instruct and Qwen2.5-Math-7B-Instruct, evaluated with greedy decoding (temperature $= 0$) and our unified answer extraction pipeline.

TTSV. *Prefix* length 20, trained for 20 epochs with AdamW optimizer, batch size 2, gradient accumulation 8 steps, and linear learning rate schedule. Learning rate is 1e-3 for Qwen-1.5B and 1e-5 for Qwen-7B. We vary the random seed controlling *prefix* initialization and training-data shuffling, using seeds 15–19 for Table 1. The content probes in Section 4.3 reuse the prefixes trained with seeds 15–17.

LTPO. Table 12 reports the per-benchmark hyperparameters. We vary the random seed controlling Gaussian perturbations during per-problem optimization, using seeds 42–46 for Table 1. The content probes in Section 4.3 reuse the prompt vectors optimized with seeds 42–44.

Table 12: LTPO hyperparameters.

Parameter	GSM8K	MATH-500	AIME24
tokens	8	8	8
steps	20	20	20
top_k	10	10	10
sigma	20	20	5
sigma_decay	0.95	0.95	0.95
lr	1e-2	5e-3	5e-2

SoftCoT. Projection module trained for 10 epochs. Thought tokens: 32 during training, 4 during evaluation. Batch size 4 for GSM8K, 1 for MATH with gradient accumulation 4. The small (assistant) model is Qwen2.5-Math-1.5B-Instruct in both configurations, while the large model is Qwen2.5-Math-1.5B-Instruct (learning rate 2e-5) or Qwen2.5-Math-7B-Instruct (learning rate 5e-6). For MATH-500 and AIME24 evaluation, we use the projection module trained on GSM8K, as it yields higher accuracy than the one trained on MATH. We vary the random seed controlling projection initialization and training-data shuffling, using seeds 42–46 for Table 1. The content probes in Section 4.3 reuse the projection modules trained with seeds 42–44.

A.4 Rejection-Sampling SFT Setup

We use Qwen2.5-Math-7B-Instruct for both data collection and the initial SFT model. The source pool contains 45,792 problems from OpenR1-Math-46k [Yan et al., 2025][3]. The fixed hard subset contains 32,737 problems with at most one correct solution among four clean rollouts. Each collection condition uses eight rollouts per hard problem at temperature 1.0. RSP uses 20 *suffix* vectors, with eight independently drawn groups for the eight rollout slots. The collection implementation samples the vectors from $\mathcal{N}(0, 0.0226^2\mathbf{I})$ and reuses each group across problems within a run. Collection seeds 42, 43, and 44 control RSP generation and decoding randomness for RSP, and decoding randomness for Baseline. Only correct trajectories are retained.

For each collected dataset, we fine-tune the same initial model for one epoch using response-only loss. We use a learning rate of 5×10^{-6}, cosine decay, a warmup ratio of 0.05, weight decay 0.01, an effective batch size of 32, and a maximum sequence length of 4,096. The SFT seed is fixed at 42, so the reported variation is across collection seeds rather than independent SFT initialization seeds. SFT uses eight NVIDIA H100 GPUs with 80GB memory each. Logged training runtimes per run are approximately 15 minutes for Baseline-SFT and 55 minutes for RSP-SFT, excluding data collection and evaluation.

Evaluation uses no RSP and a maximum generation length of 4,096 tokens. Minerva Math, GaoKao2023-EN, and OlympiadBench use greedy decoding. AIME24 and AMC23 use 32 samples per problem at temperature 1.0 and top-$p = 0.95$, reporting Avg@32. The reported average is the unweighted mean of these five benchmarks.

[3] `https://huggingface.co/datasets/Elliott/Openr1-Math-46k-8192`

A.5 Full Entropy Curves

Figure 5 shows the full normalized (0–100%) generation trajectories for entropy, top-1 probability, and varentropy, and Table 13 reports the corresponding scalar statistics including overall means, first-5% means, and the average number of generated tokens per problem. All methods converge to similar levels after the initial generation stage, confirming that the distributional changes induced by RSP are localized to the early reasoning phase.

Table 13: Per-step statistics for Qwen2.5-Math-1.5B-Instruct on MATH-500 across the full generation and the first 5% of generation steps, together with the average number of generated tokens per problem. Top-1 Prob (overall) is reported as a weighted average over the first-10%/middle/last-10% segment means with weights $0.1/0.8/0.1$. Extension of Figure 3 (main text) and Figure 5.

Metric	**Baseline**	***Prefix***	***Infix***	***Suffix***	**LTPO**	**SoftCoT**	**TTSV**
Tokens (mean)	575.7	558.4	549.9	573.3	592.3	594.4	577.4
Entropy (first 5%)	0.1332	0.1366	0.1402	0.1941	0.1662	0.4218	0.1359
Entropy (overall)	0.0871	0.0881	0.0899	0.1049	0.0909	0.1059	0.0864
Top-1 Prob (first 5%)	0.9535	0.9523	0.9525	0.9373	0.9437	0.9156	0.9534
Top-1 Prob (overall)	0.9684	0.9681	0.9678	0.9647	0.9683	0.9651	0.9685
Varentropy (first 5%)	0.2181	0.2207	0.2463	0.4010	0.2953	2.2234	0.2174
Varentropy (overall)	0.1353	0.1375	0.1422	0.2038	0.1452	0.2404	0.1361

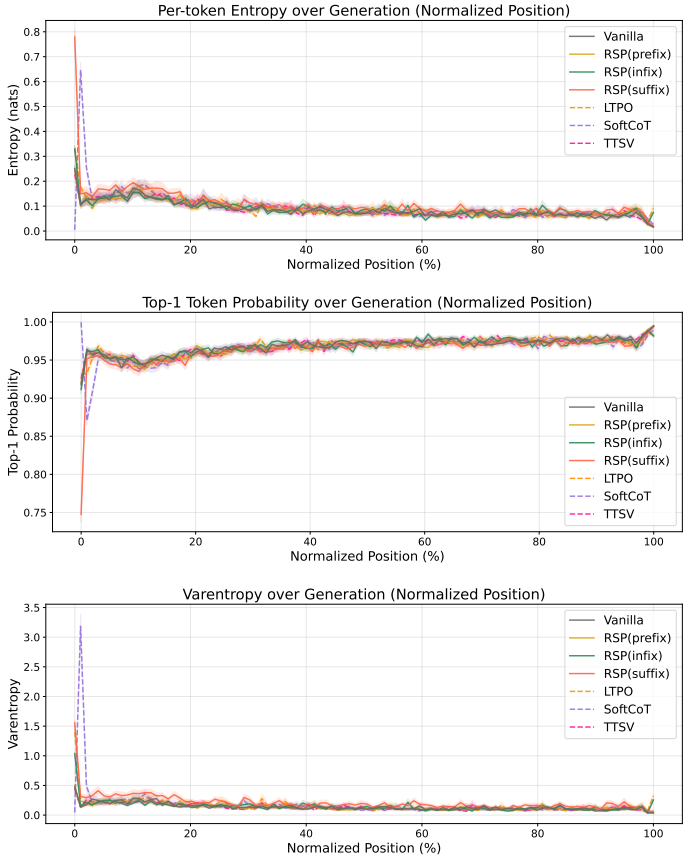


Figure 5: Mean entropy (top), top-1 probability (middle), and varentropy (bottom) over the full normalized generation trajectory (0–100%). Averaged over MATH-500, shading indicates ± 1 SEM. Solid lines: RSP variants and the baseline. Dashed lines: prior *soft prompt* methods. All methods converge after the initial stage.

B Content-Probe Replication at 7B

We replicate probes (i) and (ii) of §4.3 on Qwen2.5-Math-7B-Instruct, retaining each method's injection budget and position (20 *prefix* tokens for TTSV, 8 *infix* tokens for LTPO, and 4 *infix* tokens for SoftCoT). SoftCoT uses Qwen2.5-Math-1.5B-Instruct as its assistant. Evaluation uses greedy decoding with the `mathstral` prompt and a chat template. Results are mean accuracy and standard deviation over three seeds: 15–17 for TTSV and 42–44 for LTPO and SoftCoT. Table 14 reports the conditions corresponding to the main-text table. "50% random" denotes replacement at a randomly selected half of the injected positions.

Table 14: **Replication of the content probes at 7B.** Accuracy (%, mean ± std, three seeds) on Qwen2.5-Math-7B-Instruct. Injection length and position remain fixed within each method. The three replacement rows use RSP, and the final two rows disrupt learned structure.

Method	Intervention	MATH-500	GSM8K	AIME24
TTSV	**Trained**	83.60 ±0.43	95.02 ±0.16	15.56 ±1.57
	50% front	81.80 ±0.16	94.95 ±0.04	15.56 ±4.16
	50% random	82.67 ±1.23	95.15 ±0.21	14.44 ±3.14
	100% RSP	82.13 ±0.34	94.95 ±0.23	18.89 ±3.14
	Token-shuffle	83.47 ±1.72	94.92 ±0.12	13.33 ±2.72
	Dim-permute	82.13 ±1.98	94.82 ±0.26	14.44 ±5.67
LTPO	**Trained**	82.87 ±0.34	94.92 ±0.16	13.33 ±2.72
	50% front	81.73 ±0.25	94.79 ±0.14	15.56 ±1.57
	50% random	81.87 ±0.19	94.87 ±0.36	12.22 ±1.57
	100% RSP	82.87 ±0.57	95.00 ±0.12	11.11 ±1.57
	Token-shuffle	82.67 ±0.50	95.10 ±0.07	14.44 ±3.14
	Dim-permute	83.20 ±0.16	94.79 ±0.19	13.33 ±0.00
SoftCoT	**Trained**	83.67 ±0.34	94.57 ±0.25	17.78 ±1.57
	50% front	83.33 ±0.19	94.69 ±0.06	15.56 ±1.57
	50% random	83.07 ±0.19	94.72 ±0.04	13.33 ±0.00
	100% RSP	82.53 ±0.25	94.59 ±0.13	16.67 ±2.72
	Token-shuffle	82.93 ±0.19	94.74 ±0.09	14.44 ±1.57
	Dim-permute	83.67 ±0.68	94.97 ±0.13	15.56 ±1.57

Replacing trained vectors. At 7B, full replacement leaves LTPO's MATH-500 accuracy unchanged at 82.87%, but reduces TTSV from 83.60% to 82.13% and SoftCoT from 83.67% to 82.53%. GSM8K changes by at most 0.08 pp under full replacement across the three methods at 7B. Thus, random replacement retains much of the performance at both scales, although the 7B MATH-500 results show a modest advantage for the trained vectors in two methods.

Disrupting learned structure. At 7B, dimension permutation changes MATH-500 accuracy from 82.87% to 83.20% for LTPO and leaves SoftCoT unchanged at 83.67%, while TTSV decreases from 83.60% to 82.13%. These results show substantial retention of performance after disrupting learned structure, with method- and benchmark-dependent differences rather than uniform equivalence.

Overtraining (probe (iii)) has not been evaluated at 7B. The overtraining results in Table 5 apply only to the 1.5B model.

C Sensitivity to the RSP Distribution and Scale

§3.1 draws RSP from an isotropic Gaussian fitted to the entrywise mean and variance of the embedding table. Proposition 1 motivates this choice, since isotropy covers every direction equally under a fixed second-moment budget. Here we test whether the choice matters in practice. We vary (a) the shape of the distribution and (b) its scale and the number of injected tokens. For (a), we compare the isotropic law with two alternatives that each change one of its properties. *Diagonal*, $\mathcal{N}(\boldsymbol{\mu}_v, \mathrm{diag}(\boldsymbol{\sigma}_v^2))$ with $\boldsymbol{\mu}_v$ and $\boldsymbol{\sigma}_v$ the per-dimension mean and standard deviation of $\mathbf{W}_E$, follows the anisotropy of the embedding table. *Sphere* draws a direction uniformly on the unit sphere and fixes the norm to the mean token-embedding norm, which removes the randomness in length. All runs use Qwen2.5-Math-Instruct with *suffix* injection and greedy decoding, and report the mean and standard deviation over five RSP seeds.

Table 15: **Choice of distribution.** Accuracy (%, mean ± std, five seeds) with $L = 20$ *suffix* tokens. *Isotropic* is the RSP law of Eq. (1). *Diagonal* and *Sphere* are defined in the text above. The baseline uses no RSP and is deterministic.

Model	Distribution	GSM8K	MATH-500	AIME24
Qwen2.5-Math-1.5B-Instruct	Baseline (no RSP)	85.29	73.00	10.00
	Isotropic (RSP)	84.99 ±0.22	74.12 ±1.36	13.33 ±5.27
	Diagonal	85.29 ±0.39	73.12 ±0.69	9.33 ±2.49
	Sphere	84.94 ±0.20	73.56 ±0.56	12.00 ±4.00
Qwen2.5-Math-7B-Instruct	Baseline (no RSP)	95.45	83.20	13.33
	Isotropic (RSP)	95.60 ±0.08	83.52 ±0.56	16.67 ±0.00
	Diagonal	95.83 ±0.17	82.84 ±0.48	15.33 ±1.64
	Sphere	95.65 ±0.12	83.48 ±0.32	14.67 ±1.64

Table 16: **Scale and token count.** MATH-500 accuracy (%, mean ± std, five seeds) on Qwen2.5-Math-1.5B-Instruct with *suffix* injection. Each row sweeps one axis and keeps the others at the default ($\alpha = \beta = 1$, $L = 20$). α multiplies σ_E. β rescales each vector to β times the mean token-embedding norm. Each sweep was run independently, so the default setting appears with slightly different values across rows. The no-RSP baseline is 73.00.

σ scale α	0.25	0.5	1	2	4		
MATH-500	73.88 ±1.27	73.92 ±0.84	74.32 ±1.36	73.56 ±0.68	73.84 ±0.55		
Norm scale β	0.25	0.5	1	2	4		
MATH-500	73.92 ±0.52	73.80 ±0.55	74.24 ±0.79	73.52 ±0.59	74.16 ±0.52		
Tokens L	1	4	8	16	32	64	128
MATH-500	73.72 ±0.70	74.48 ±0.50	73.12 ±0.72	74.28 ±1.21	74.48 ±1.20	72.92 ±0.92	71.08 ±1.03

Choice of distribution. Isotropy has the highest mean accuracy on MATH-500 and AIME24 at both scales, while Diagonal is highest on GSM8K (Table 15). The gaps are small. On MATH-500 the three distributions span 73.12–74.12% at 1.5B and 82.84–83.52% at 7B, and on GSM8K they span 84.94–85.29% and 95.60–95.83%. Each gap is smaller than the seed standard deviation. AIME24 varies more (9.33–13.33% at 1.5B), but its 30 problems move accuracy in steps of 3.33 points and the standard deviations reach 5.27 points. The table therefore shows at most a slight edge for isotropy and no evidence that another distribution would be preferable. This agrees with §4.3, where the effect depends on injecting vectors of a suitable magnitude and not on their precise content.

Scale and token count. Scaling σ_E or the norm by any factor between 0.25 and 4, a sixteen-fold range, keeps MATH-500 accuracy within 73.52–74.32%, again inside one standard deviation (Table 16). The token count behaves the same way up to $L = 32$ (73.12–74.48%). Beyond that, accuracy declines to 72.92% at $L = 64$ and 71.08% at $L = 128$, which falls below the no-RSP baseline of 73.00%. A long random prompt therefore starts to hurt. This is consistent with Theorem 1, where a larger L raises the attention mass on random tokens and delays the dilution that lets the response commit to a single completion.

D Token-Level Distribution Metrics

This appendix provides formal definitions for the metrics used in Section 4.5. Numerical results are reported in Table 6 of the main text. We extract first-token logits via a single forward pass (no autoregressive generation) and store the top-100 token ids and logit values per problem. Temperature conditions reuse the baseline logits scaled by $1/\tau$, requiring no separate inference. RSP draws 16 independent seeds per problem (MATH-500), and reported metrics are the mean across the 16 seeds. Pass@1 accuracy is computed over 16 samples per problem, with the baseline at τ=1.0 reaching $73.92 \pm 0.78\%$. Let P_{base} denote the baseline next-token distribution at τ=1.0 and P_{target} the candidate distribution being compared (e.g., baselines at τ=2.0, 3.0 or RSP at τ=1.0). All metrics are computed analytically from saved logits at the first generation step.

Spearman rank correlation. Spearman's ρ is the Pearson correlation coefficient between the token ranks of two distributions:

$$\rho \;=\; \mathrm{Pearson}(\mathrm{rank}(P_{\text{target}}),\ \mathrm{rank}(P_{\text{base}}))\,. \tag{4}$$

Because temperature scaling $P^{(\tau)} \propto P^{1/\tau}$ is a strictly monotone transform, the rank order of tokens is preserved exactly, so $\rho = 1$ for any $\tau > 0$ as a mathematical identity. Any value $\rho < 1$ therefore indicates rank reorganization beyond what temperature scaling can produce.

Mass outside top-K. Let $\mathrm{TopK}(P_{\text{base}})$ be the set of K tokens with the highest probability under P_{base}. The probability mass that the candidate distribution places *outside* this set is

$$\mathrm{MassOutside}_K \;=\; 1 - \sum_{t \in \mathrm{TopK}(P_{\text{base}})} P_{\text{target}}(t). \tag{5}$$

This directly measures support expansion: how much probability the candidate assigns to tokens that are not in the baseline's preferred set. Reported values use $K = 10$.

Jensen–Shannon divergence. The symmetrized, bounded version of Kullback–Leibler divergence:

$$\mathrm{JS}(P, Q) \;=\; \tfrac{1}{2}\,\mathrm{KL}(P \,\|\, M) + \tfrac{1}{2}\,\mathrm{KL}(Q \,\|\, M), \qquad M = \tfrac{1}{2}(P + Q). \tag{6}$$

We use base-2 logarithms so that JS $\in [0, 1]$. Tokens absent from the stored top-100 are assigned a sentinel logit (-10^9) before softmax, which is negligible for our setting.

E Comparison with Intermediate and Annealed Temperatures

Figure 4 compares RSP only with temperature sampling at τ=1.0. Table 6 shows that raising the temperature to τ=3.0, which matches RSP's distributional shift in magnitude, collapses accuracy. That value was chosen to match magnitude, not to maximize Pass@N. Here we extend Figure 4 to milder and time-varying temperatures and ask whether either can reach RSP's outcome diversity. We compare (a) constant temperatures $\tau \in \{1.0, 1.2, 1.5, 1.8, 2.0\}$ and (b) annealed schedules that start at $\tau_s \in \{2.0, 2.5, 3.0\}$ and decrease to 1.0, either linearly or along a cosine curve. A schedule gives high diversity at the first tokens and then lets the response settle, which mimics the early-then-fading influence of RSP. The annealing length T is the mean generation length at τ=1.0 (547 tokens on MATH-500 and 913 on AIME24), so the schedule reaches 1.0 within an average completion. We also include (c) the trained soft prompts as a reference. All runs follow the setting of Figure 4 (Qwen2.5-Math-1.5B-Instruct, 16 samples per problem, top-p=0.95).

Constant temperature. Raising the temperature does not raise Pass@N (Table 17). On MATH-500, Pass@16 is highest at τ=1.0 (91.80%) and decreases as τ grows. On AIME24, τ=1.2 and 1.5 lift Pass@16 from 20.00% to 23.33%, one additional problem out of 30. From τ=1.8 onward, accuracy collapses on both benchmarks.

Annealed temperature. Annealing does not help either. Every schedule falls below constant τ=1.0, and on AIME24 no schedule exceeds 3.33% at k=16. Even the mildest schedule (linear, τ_s=2.0) reaches only 73.60% on MATH-500 at k=16, below constant τ=1.8 (82.40%) although it is cooler for all but the first $\sim$110 tokens. This suggests that the high-temperature tokens at the start already derail the solution and later cooling cannot recover it. The pattern agrees with our view that the first decoding steps decide the trajectory. RSP acts on the same early steps but keeps single-sample accuracy, because it reranks tokens instead of flattening the whole distribution (Table 6).

Comparison with RSP. RSP gives the highest Pass@N on AIME24 at every k, reaching 36.67% at k=16 against 23.33% for the best temperature setting and 30.00% for the best soft prompt. On MATH-500, all non-collapsed methods sit near the ceiling (90.4–92.8% at k=16). RSP is highest at k=8 and k=16, while LTPO is slightly higher at k=4. Temperature only flattens the distribution and preserves the token ranking (Table 6), whereas RSP changes the ranking. Neither a milder nor a time-varying temperature closes this gap.

Table 17: **Pass@k under intermediate and annealed temperatures.** Qwen2.5-Math-1.5B-Instruct, 16 samples per problem. Annealed schedules start at τ_s and reach τ=1.0 after T tokens. Soft prompts and RSP are sampled at τ=1.0. Bold marks the highest value in each column.

		MATH-500			AIME24		
Category	**Setting**	k=4	k=8	k=16	k=4	k=8	k=16
(a) Constant τ	τ=1.0	85.09	88.80	91.80	17.63	19.21	20.00
	τ=1.2	84.92	88.85	91.60	15.21	19.08	23.33
	τ=1.5	82.32	87.09	90.40	13.63	18.32	23.33
	τ=1.8	66.44	75.25	82.40	4.45	5.88	6.67
	τ=2.0	45.86	58.22	68.40	1.67	3.33	6.67
(b) Linear anneal	τ_s=2.0	49.09	63.04	73.60	0.00	0.00	0.00
	τ_s=2.5	20.16	30.98	43.80	0.83	1.67	3.33
	τ_s=3.0	15.68	25.12	36.20	0.00	0.00	0.00
(b) Cosine anneal	τ_s=2.0	44.59	59.23	72.00	0.83	1.67	3.33
	τ_s=2.5	20.07	31.18	43.40	0.83	1.67	3.33
	τ_s=3.0	17.61	28.09	40.80	0.00	0.00	0.00
(c) Soft prompts	LTPO	**85.73**	89.33	91.60	15.51	19.17	23.33
	SoftCoT	84.59	88.79	91.40	18.20	23.00	30.00
	TTSV	84.59	88.43	91.40	15.43	17.96	20.00
RSP (*suffix*, independent draws)		85.58	**89.58**	**92.80**	**21.36**	**27.67**	**36.67**

F Qualitative Analysis: How RSP Reshapes Reasoning Trajectories

This appendix complements the outcome-level Pass@N results of §4.5 with a qualitative case study on AIME24, using the same N=16 sampling budget. We compare two of the four conditions defined in §4.5: *Baseline* (condition (i), τ=1.0) and *RSP (indep. seed)* (condition (iv), τ=1.0, *suffix*, L=20). Setting: Qwen2.5-Math-1.5B-Instruct, 30 AIME24 problems, 16 samples per problem per condition.

F.1 Aggregate Pattern Frequencies

Table 18 tabulates how RSP shifts coverage relative to Baseline across the 30 AIME24 problems.

Table 18: Coverage patterns across 30 AIME24 problems, N=16 samples per condition.

Pattern	**Frequency**	**Problems**
RSP-only correct (Baseline=0/16, RSP≥1/16)	5/30 (16.7%)	#6, #8, #14, #22, #25
Baseline-only correct (Baseline≥1/16, RSP=0/16)	0/30 (0%)	—
Both correct	6/30 (20.0%)	—
Both wrong	19/30 (63.3%)	—

Within the N=16 budget, RSP produces strict gains on five problems and never causes a strict regression. The two case studies below illustrate two faces of the same phenomenon: divergent solution framing (#6, §F.2) and divergent structural assumption (#22, §F.3). In both cases, RSP and Baseline diverge already in the opening sentences of the trajectory, and the divergence determines whether the trajectory reaches the correct answer.

F.2 Case Study 1: AIME24 Problem 6 (Constrained Optimization)

Problem.

> Let $\mathcal{B}$ be the set of rectangular boxes with surface area 54 and volume 23. Let r be the radius of the smallest sphere that can contain each of the rectangular boxes that are elements of $\mathcal{B}$. The value of r^2 can be written as $\frac{p}{q}$, where p and q are relatively prime positive integers. Find $p+q$.

Ground truth: $\boxed{721}$, with the optimal box at $a=b=2$, $c=23/4$ giving $r^2=657/64$.

Aggregate result. Baseline produces 0/16 correct and RSP produces 1/16 correct. Baseline's 16 wrong final answers span 10 distinct values, with the modal answer 31 appearing in 4/16 samples,

and the correct value 721 does not appear. RSP's 16 samples produce 11 distinct values and include 721 exactly once.

Failure mode (Baseline). Baseline's representative trajectory (final answer 49) opens with an *algebraic identity* frame and never solves for the actual box dimensions.

Baseline trajectory (final answer 49) — excerpts

Opening: To solve the problem, we start by defining the dimensions of a rectangular box as a, b, and c. The surface area and volume of the box are given by $2(ab + bc + ca) = 54$ and $abc = 23$. We need to find the radius r of the smallest sphere that can contain the box, given by $r = \frac{1}{2}\sqrt{a^2 + b^2 + c^2}$. First, we need to find the expression for $a^2 + b^2 + c^2$ in terms of the given equations.
Identity manipulation: Squaring the surface area equation yields $(ab + bc + ca)^2 = 2916$, which expands to $a^2b^2 + b^2c^2 + c^2a^2 + 2abc(a + b + c) = 2916$. Substituting $2abc = 46$ and $(a + b + c)^2 = x + 54$ where $x := a^2 + b^2 + c^2$, then attempting to express $a^4 + b^4 + c^4$ in terms of x, the manipulation collapses to the tautology $x^2 = x^2 - 2y + 2y$.
Termination by guess: "This equation is always true, so we need another way to find x. ... After solving, we find $x = 45$." Thus $r^2 = 45/4$, and $p + q = 45 + 4 = \boxed{49}$.

Diagnosis. The trajectory frames the task as *find the closed form of $a^2 + b^2 + c^2$ from the given constraints* and never attempts to solve for the actual dimensions. Two consecutive identity manipulations collapse to a tautology that does not yield x, after which the trajectory commits to $x = 45$ without justification ("After solving, we find"). This pattern — identity manipulation, tautology, terminal guess — recurs across Baseline's 16 samples. The constraint set under-determines $a^2 + b^2 + c^2$ in closed form, and the algebraic-identity framing has no path to the correct value.

Success mode (RSP). The correct RSP trajectory (final answer 721) opens with an *optimization* frame and a symmetry assumption that reduces the system to a single cubic.

RSP trajectory (final answer 721) — excerpts

Opening: To solve the problem, we need to find the dimensions of the rectangular box that *minimizes* the radius of the smallest sphere that can contain it, given the surface area and volume constraints. The radius is $r = \frac{1}{2}\sqrt{a^2 + b^2 + c^2}$. To minimize r, we minimize $a^2 + b^2 + c^2$.
Symmetry reduction: A more straightforward approach is to use symmetry: $a^2 + b^2 + c^2$ is minimized when a, b, c are as close to each other as possible. Assume $a = b$ and solve for c: $2a^2 + 2ac = 27$ and $a^2c = 23$. From $a^2c = 23$, $c = 23/a^2$. Substituting: $2a^2 + 46/a = 27$, multiplying through by a gives $2a^3 - 27a + 46 = 0$.
Rational root and termination: By the Rational Root Theorem, $a = 2$ is a root. Then $c = 23/4$, $a^2 + b^2 + c^2 = 4 + 4 + 529/16 = 657/16$, $r^2 = 657/64$, and $p + q = \boxed{721}$.

Diagnosis. Three properties of the opening drive the success: **(i)** the task is framed as *minimizing r* (and thus $a^2 + b^2 + c^2$) rather than as a closed-form expression hunt, **(ii)** the path is committed to solving for a, b, c instead of for $a^2 + b^2 + c^2$ alone, and **(iii)** a symmetry assumption $a = b$ reduces the system to one cubic in one unknown. The cubic $2a^3 - 27a + 46 = 0$ has $a = 2$ as a rational root, leading to the explicit configuration $(2, 2, 23/4)$ and the correct $r^2 = 657/64$.

Interpretation. Baseline's failure is rooted in the *opening sentence*: framing the task as "find the expression for $a^2 + b^2 + c^2$ from the given equations" commits the trajectory to a path that does not have a closed-form solution under the given constraints. RSP's perturbation produces, in 1 of 16 attempts, a trajectory whose opening frame is *constrained optimization* with a symmetry-based variable reduction. The two trajectories share the same model weights, the same problem constraints, and the same sampling temperature. The input-side RSP perturbation is the only intervention, and its visible effect is the different framing the model commits to in the first few sentences.

F.3 Case Study 2: AIME24 Problem 22 (Constrained List Construction)

Problem.

A list of positive integers has the following properties:

- The sum of the items in the list is 30.

- The unique mode of the list is 9.
- The median of the list is a positive integer that does not appear in the list itself.

Find the sum of the squares of all the items in the list.

Ground truth: $\boxed{236}$, corresponding to $\{5, 7, 9, 9\}$ with median $\frac{7+9}{2} = 8$.

Aggregate result. Baseline produces 0/16 correct and RSP produces 3/16 correct. Baseline's 16 samples produce 16 distinct final answers (no two agree), and the correct value 236 does not appear. RSP's 16 samples concentrate slightly more, with 236 (3 times) and 248 (3 times) tied as modal answers and the remaining mass spread across 10 distinct values.

Structural-assumption split. The constraint "the median is a positive integer that does not appear in the list" *requires* an even-length list: for an odd-length list the median is by definition an element of the list. Across Baseline's 16 samples, 6 commit to an odd-length list (typically 5 elements) at the opening and only 3 consider an even-length list. Across RSP's 16 samples, 8 commit to odd and 6 consider even. The shift in the even-length consideration rate (3/16 $\rightarrow$ 6/16) is the structural correlate of RSP's accuracy gain.

Failure mode (Baseline). Baseline's representative trajectory (final answer 224) commits to an odd-length list *at the opening* and never reconsiders.

Baseline trajectory (final answer 224) — excerpts

Opening commitment: "Let's start by assuming the list has an odd number of elements. If the list has $2k + 1$ elements, the median is the $(k + 1)$-th element. Let's try with the smallest possible odd number…"
Construction: 5 elements, $a_1 + a_2 + 9 + a_4 + 9 = 30$ so $a_1 + a_2 + a_4 = 12$. Try $a_1 = 2$, $a_2 = 3$, $a_4 = 7$: list $\{2, 3, 7, 9, 9\}$, median 7, "which does not appear in the list, so this is a valid candidate."
Termination: Sum of squares $= 4 + 9 + 49 + 81 + 81 = \boxed{224}$.

Diagnosis. Two errors compound: (i) the trajectory *defaults to odd length* at the opening without justification, and the constraint "median not in the list" forces even length, which the trajectory never derives, and (ii) the median value 7 is verified as a positive integer but *not* verified as absent from the list (7 is in fact an element). The same structural assumption recurs in several Baseline samples. The elements vary, but the odd-length commitment and incomplete constraint check remain.

Success mode (RSP). The correct RSP trajectory (final answer 236) considers $n = 4$ from the outset and treats the median formula by parity.

RSP trajectory (final answer 236) — excerpts

Opening (Step 2: median by parity). "If the list has an odd number of elements, the median is the middle element. If the list has an even number of elements, the median is the average of the two middle elements. Since the median is a positive integer that does not appear in the list, it must be an integer and not equal to any of the other numbers in the list."
Step 3: starting at $n = 4$. "$n \geq 4$ (at least two 9's). Case 1: $n = 4$. Sum: $9 + 9 + x + y = 30$, so $x + y = 12$."
Construction: Try $x = 5$, $y = 7$: list $\{5, 7, 9, 9\}$, median $\frac{7+9}{2} = 8$ (integer, not in list). All conditions satisfied, so the sum of squares $= 25 + 49 + 81 + 81 = \boxed{236}$.

Diagnosis. Three properties distinguish this trajectory: **(i)** *even length is considered first* ($n = 4$ enumerated as the initial case), **(ii)** the median is computed as an arithmetic mean for even length, producing $8 \notin \{5, 7, 9, 9\}$, and **(iii)** the constraint check is performed against the candidate *value* (8), not against the structural position. The other two correct RSP samples reach 236 through partially incorrect intermediate reasoning, suggesting that some RSP samples that begin with the same odd-length assumption as Baseline can still arrive at the correct numerical answer.

Interpretation. Baseline's failure is rooted in the *first structural assumption* after the conditions are listed: the trajectory commits to an odd-length list within its first few hundred tokens and never reconsiders. RSP's perturbation increases the fraction of samples that consider $n = 4$ from 3/16 to 6/16, and one of these reaches the correct constraint check. The split between odd-length and

even-length openings is the empirical fingerprint of the template-selection shift. On a small fraction of samples, the alternative template happens to be the structurally consistent one.

F.4 Summary of Case Studies

Both case studies exhibit the same shape. Baseline's 16 samples cluster around a single failure template (algebraic-identity framing for #6 and odd-length-list commitment for #22) and never break out of it within the sample budget. RSP does not change the model's underlying knowledge: most RSP samples fall into the same templates as Baseline. What RSP changes is *which template the model commits to in the opening sentences*. On a small fraction of trajectories, the alternative template happens to admit a solvable subproblem (the optimization framing for #6, the even-length list for #22), and the trajectory reaches the correct answer. The accuracy lift (0/16 $\rightarrow$ 1/16 on #6, 0/16 $\rightarrow$ 3/16 on #22) is the outcome-level signature of this opening-sentence divergence, consistent with the input-side branch-opening mechanism of §3.2, where RSP can alter early decoding decisions before the trajectory commits to a structural template.

G Prompt-Law Properties Used by the Theory

This appendix isolates the only properties of the RSP law used in the main text: centeredness, direction-agnostic covariance under a variance budget, and positive probability on every nonempty open set. We avoid stronger semantic claims because Section 3.2 needs only these geometric and measure-theoretic facts.

G.1 Centered Perturbations Do Not Impose Systematic First-Order Drift

Let z_a^{Base} denote the no-RSP output logit at vocab token a and $z_a^{\text{RSP}}(\bar{h})$ the corresponding logit when the centered RSP perturbation takes value $\bar{h}$. The (true) vocab-logit gap under RSP is $\Delta_{ab}(\bar{h}) := z_a^{\text{RSP}}(\bar{h}) - z_b^{\text{RSP}}(\bar{h})$, distinct from the no-RSP baseline gap $\Delta_{ab}^{\text{Base}} := z_a^{\text{Base}} - z_b^{\text{Base}}$. Note that $\bar{h} = 0$ corresponds to the centered RSP at its entrywise mean rather than to the no-RSP forward pass, so in general $\Delta_{ab}(0) \neq \Delta_{ab}^{\text{Base}}$. The transformer's logit map is non-linear in $\bar{h}$, but a first-order Taylor expansion around $\bar{h} = 0$ yields the local surrogate

$$\Delta_{ab}^{\text{lin}}(\bar{h}) \;:=\; \Delta_{ab} + b_{ab}^{\top}\bar{h},$$

with $\Delta_{ab} := \Delta_{ab}(0)$ and $b_{ab} := \nabla_{\bar{h}}(z_a^{\text{RSP}} - z_b^{\text{RSP}})|_{\bar{h}=0}$. The proposition below is stated for the surrogate Δ_{ab}^{lin}. For the true non-linear gap $\Delta_{ab}(\bar{h})$, the first-order term has zero mean under centered perturbations, but the Taylor remainder $r_{ab}(\bar{h}) = O(\|\bar{h}\|^2)$ may contribute a second-order mean shift. We make no claim about its sign or magnitude.

Proposition 2 (No systematic first-order drift in the surrogate)**.** *If $\mathbb{E}[\bar{h}] = 0$, then*

$$\mathbb{E}\big[\Delta_{ab}^{\text{lin}}(\bar{h})\big] = \Delta_{ab}$$

for every token pair (a, b). If a deterministic centered prompt $\bar{h}_0$ is reused across runs, then

$$\Delta_{ab}^{\text{lin}}(\bar{h}_0) - \Delta_{ab} = b_{ab}^{\top}\bar{h}_0$$

is a fixed directional bias.

Proof. The centered case is immediate from linearity of expectation:

$$\mathbb{E}\big[\Delta_{ab}^{\text{lin}}(\bar{h})\big] = \Delta_{ab} + b_{ab}^{\top}\mathbb{E}[\bar{h}] = \Delta_{ab}.$$

The deterministic case follows by substitution. □

This proposition rules out a run-averaged first-order bias. It does not say that individual prompt draws leave the logits unchanged.

G.2 Proof of Proposition 1

For a unit vector u, the first-order perturbation energy available along u is $\mathrm{Var}_{h\sim D}(u^\top h) = u^\top \Sigma_D u$. The worst-case directional energy is therefore the minimum Rayleigh quotient of Σ_D.

Proof. For any symmetric covariance matrix Σ_D,

$$\min_{\|u\|=1} u^\top \Sigma_D u = \lambda_{\min}(\Sigma_D).$$

Let the eigenvalues of Σ_D be $\lambda_1, \ldots, \lambda_d$. Then

$$\lambda_{\min}(\Sigma_D) \le \frac{1}{d}\sum_{m=1}^{d} \lambda_m = \frac{\mathrm{tr}(\Sigma_D)}{d} \le \frac{\rho^2}{d}.$$

Equality holds if and only if all eigenvalues are equal, i.e., $\Sigma_D = (\rho^2/d)\mathbf{I}_d$. □

The proof uses only covariance. We choose a Gaussian law for RSP because it adds the open-set reachability property proved next.

G.3 Open-Set Reachability of Isotropic Gaussian Prompts

Proposition 3 (Open-set reachability)**.** *Let $\bar{h} \sim \mathcal{N}(0, \sigma^2\mathbf{I}_d)$ with $\sigma > 0$. Then every nonempty open set $U \subseteq \mathbb{R}^d$ satisfies*

$$\Pr(\bar{h} \in U) > 0.$$

Proof. The Gaussian density

$$(2\pi\sigma^2)^{-d/2} \exp\left(-\frac{\|\bar{h}\|^2}{2\sigma^2}\right)$$

is strictly positive at every point of $\mathbb{R}^d$. Every nonempty open set contains a ball of positive Lebesgue measure, and integrating a strictly positive density over that ball gives positive probability. □

This is the only support property used in the informal branching argument of §3.2 and in Appendix H.4.3.

H Proofs for the Transformer-Level Mechanism

This appendix follows the same order as the main theory section: exploration, annealing, and performance gain. The prompt-law facts used before these proofs are collected in Appendix G. The attention results below, Eq. (2) and Theorem 1, analyze one head at a time. They are a local diagnosis of how attention routes the RSP contribution, not a complete causal account of how the perturbation propagates through the full Transformer.

H.1 Exploration: attention decomposition and perturbation size

Proposition 4 (Attention decomposition)**.** *For one attention head at query position i in layer ℓ, let $\alpha_{ij}^{(\ell)}$ be the attention weights over $n \ge 1$ unmasked real tokens and $L \ge 1$ random tokens, with finite attention logits. Define*

$$w_{r,i}^{(\ell)} := \sum_{j=1}^{L} \alpha_{i,n+j}^{(\ell)}$$

and, for $j \in [n]$,

$$\tilde{\alpha}_{ij}^{(\ell)} := \frac{\alpha_{ij}^{(\ell)}}{1 - w_{r,i}^{(\ell)}}.$$

Then

$$\mathbf{o}_i^{(\ell)} = (1 - w_{r,i}^{(\ell)})\, \tilde{\mathbf{o}}_i^{(\ell)} + \boldsymbol{\eta}_i^{(\ell)},$$

where

$$\tilde{\mathbf{o}}_i^{(\ell)} := \sum_{j=1}^{n} \tilde{\alpha}_{ij}^{(\ell)} \mathbf{v}_j^{(\ell)} \qquad \textit{and} \qquad \boldsymbol{\eta}_i^{(\ell)} := \sum_{j=1}^{L} \alpha_{i,n+j}^{(\ell)} \mathbf{v}_{r_j}^{(\ell)}.$$

Proof. The head output is

$$\mathbf{o}_i^{(\ell)} = \sum_{j=1}^{n} \alpha_{ij}^{(\ell)} \mathbf{v}_j^{(\ell)} + \sum_{j=1}^{L} \alpha_{i,n+j}^{(\ell)} \mathbf{v}_{r_j}^{(\ell)}.$$

By the softmax positivity and the existence of at least one unmasked real token, $\sum_{j=1}^{n} \alpha_{ij}^{(\ell)} = 1 - w_{r,i}^{(\ell)} > 0$, so

$$\sum_{j=1}^{n} \alpha_{ij}^{(\ell)} \mathbf{v}_j^{(\ell)} = (1 - w_{r,i}^{(\ell)}) \sum_{j=1}^{n} \frac{\alpha_{ij}^{(\ell)}}{1 - w_{r,i}^{(\ell)}} \mathbf{v}_j^{(\ell)} = (1 - w_{r,i}^{(\ell)})\, \tilde{\mathbf{o}}_i^{(\ell)}.$$

Substituting this identity gives the decomposition. □

Corollary 1 (Magnitude scales with random-token attention)**.** *For every realization of the random values,*

$$\|\boldsymbol{\eta}_i^{(\ell)}\| \le w_{r,i}^{(\ell)} \max_{j \in [L]} \|\mathbf{v}_{r_j}^{(\ell)}\|.$$

Proof. By the triangle inequality,

$$\|\boldsymbol{\eta}_i^{(\ell)}\| = \left\| \sum_{j=1}^{L} \alpha_{i,n+j}^{(\ell)} \mathbf{v}_{r_j}^{(\ell)} \right\| \le \sum_{j=1}^{L} \alpha_{i,n+j}^{(\ell)} \|\mathbf{v}_{r_j}^{(\ell)}\| \le w_{r,i}^{(\ell)} \max_{j \in [L]} \|\mathbf{v}_{r_j}^{(\ell)}\|.$$

□

Corollary 1 bounds the norm of the direct RSP contribution. This bound is distinct from the attention-mass identity in Theorem 1. No independence assumption between attention weights and random keys is required. For bounded RSP value norms, the upper bound on the random contribution decreases with $w_{r,i}^{(\ell)}$.

H.2 Corollaries of Theorem 1 on KV cache growth

Theorem 1 expresses the attention mass as $w_{r,i}^{(\ell)} = L/(L + ne^{g_i^{(\ell)}})$, with a gap $g_i^{(\ell)}$ that may vary throughout generation. Thus, the decrease of RSP attention depends on the product $ne^{g_i^{(\ell)}}$, rather than on cache growth alone. The following corollaries state the conditions for attention decay and, separately, for the direct RSP contribution to vanish.

Corollary 2 (Sequence-wise annealing)**.** *For fixed L, RSP attention decreases whenever $ne^{g_i^{(\ell)}}$ increases. If $ne^{g_i^{(\ell)}} \to \infty$, then $w_{r,i}^{(\ell)} \to 0$.*

Proof. Both statements follow directly from $w_{r,i}^{(\ell)} = L/(L + ne^{g_i^{(\ell)}})$. The condition permits a varying gap throughout generation. □

Corollary 3 (Vanishing random contribution under annealing)**.** *For fixed L, if $ne^{g_i^{(\ell)}} \to \infty$ and the RSP value norms remain uniformly bounded, then $\|\boldsymbol{\eta}_i^{(\ell)}\| \to 0$.*

Proof. Corollary 2 gives $w_{r,i}^{(\ell)} \to 0$. Corollary 1 then gives the result under the bounded-value assumption, which holds for fixed, finite RSP values in the causal KV cache. □

H.3 Attention-logit gap along generation

Under the same setting as Figure 2 (Qwen2.5-Math-7B, *suffix*, MATH-500), Table 19 reports the token-count-normalized log-sum-exp gap averaged over heads and samples. We interpret changes within each layer band along generation. The gap increases most clearly in the shallowest and deepest bands, while the middle bands show smaller and non-monotonic changes. These measurements complement the attention heatmaps. Equation (3) holds for each head and query, so substituting the mean gap into it does not in general recover the mean attention mass.

Table 19: Mean token-count-normalized log-sum-exp gap by layer depth and generation position.

Layer depth (%)	Generation position (%)				
	0–20	20–40	40–60	60–80	80–100
0–20	2.56	3.76	4.66	5.69	6.74
20–40	2.45	3.00	3.16	3.30	3.26
40–60	3.47	3.80	3.63	3.53	3.17
60–80	3.70	4.00	3.93	3.89	3.59
80–100	4.34	6.07	7.65	8.98	9.75

H.4 Performance gain: from local admission to Pass@N

From here we work at the vocab-logit and task levels, dropping the per-layer index ℓ. Section 3.2 uses a local first-order surrogate to analyze branch opening. Section 3.3 connects independent rollouts to Pass@N through assumptions (M1) and (M2). The branch-opening argument below uses the following support condition on the centered prompt law D:

$$\text{(S1)} \qquad D(U) > 0 \quad \text{for every nonempty open set } U \subseteq \mathbb{R}^d.$$

Proposition 3 shows that the isotropic Gaussian law of §3.2 satisfies (S1).

H.4.1 Autoregressive Formulation

In autoregressive decoding, the joint distribution over a sequence $y_{1:T}$ factorizes as

$$\Pr(y_{1:T} \mid x) = \prod_{t=1}^{T} p(y_t \mid x, y_{<t}),$$

where each step is governed by a prefix-conditioned distribution. Any perturbation introduced by RSP affects generation through a sequence of local changes to $p(y_t \mid x, y_{<t})$, rather than a single global distribution.

H.4.2 Local Linearization of Logits under RSP

Throughout this subsection, $\bar{h} \in \mathbb{R}^d$ denotes *one* centered prompt vector from the RSP definition (Eq. (1) in §3.1), treated as a per-token surrogate. The actual implementation uses a length-L prompt $\mathbf{H} = (h_1, \ldots, h_L)$ with L independent draws, and the arguments below isolate how one local perturbation can shift the logits. We model the perturbed logits as a function $z_a(\bar{h})$ at vocab token a, assuming local smoothness in a neighborhood of $\bar{h} = 0$:

$$z_a(\bar{h}) = z_a(0) + \nabla_{\bar{h}} z_a(0)^\top \bar{h} + r_a(\bar{h}),$$

where $r_a(\bar{h}) = O(\|\bar{h}\|^2)$. Defining $c_a := \nabla_{\bar{h}} z_a(0)$ and $z_a := z_a(0)$, we obtain the local affine surrogate

$$z_a^{\mathrm{lin}}(\bar{h}) := z_a + c_a^\top \bar{h}.$$

This approximation is used as a first-order surrogate for branch opening, not as an exact global model of the transformer logits.

H.4.3 Top-K entry region (Remark)

This complements the informal branching argument in §3.2. Define the entry region

$$\mathcal{G}_{a,K} \;:=\; \big\{\bar{h} \in \mathbb{R}^d : \#\{b \neq a : z_b^{\mathrm{lin}}(\bar{h}) > z_a^{\mathrm{lin}}(\bar{h})\} \leq K - 1\big\}.$$

If $\mathcal{G}_{a,K}$ contains a nonempty open set U, then by (S1) $D(U) > 0$, and $U \subseteq \mathcal{G}_{a,K}$ gives $\Pr_{\bar{h}}(\bar{h} \in \mathcal{G}_{a,K}) \geq D(U) > 0$. A useful sufficient condition is the existence of some $\bar{h}_0$ at which token a strictly outranks all but at most $K-1$ other tokens in the affine surrogate. By continuity, a neighborhood of $\bar{h}_0$ is then contained in $\mathcal{G}_{a,K}$. This is a rank-changing event unreachable by the monotone temperature transform, but it does not by itself imply (M1).

H.4.4 Pass@N ensemble bound (Remark)

The bound $\Pr(\text{Pass@}N \text{ on } x) \geq 1 - (1 - p_{\min}(x))^N$ in the main text is a standard independent-Bernoulli union argument. Let A_n be the event that the n-th run reaches a correct trajectory on x. Under (M1) $\Pr(A_n) \geq p_{\min}(x)$, and under (M2)

$$\Pr\Big(\bigcap_{n=1}^{N} A_n^c\Big) \leq (1 - p_{\min}(x))^N, \qquad \Pr\Big(\bigcup_{n=1}^{N} A_n\Big) \geq 1 - (1 - p_{\min}(x))^N \;\geq\; 1 - e^{-N p_{\min}(x)}.$$

This is a generic fact, not specific to RSP. RSP's role is to make branch opening possible and to support (M2) through independent resampling, while whether a branch satisfies the task-side correctness condition (M1) remains task-dependent.

I Attention Mass Analysis

This appendix formalizes the per-token attention mass reported in Figure 2 and the exclusion criteria applied to the reasoning span and the layer axis. For each of the 500 MATH-500 problems, a fresh RSP $\mathbf{H} \in \mathbb{R}^{L \times d}$ with $L = 10$ is sampled independently, so the reported heatmaps average over both problem variability and RSP-vector variability.

I.1 Per-Token Attention Mass

For each problem, we measure attention on the concatenated sequence [prompt; $\mathbf{H}$; generation], where the generation is the trajectory produced by the same model under matching RSP injection. Let $\alpha_{i,j}^{(\ell,m)}$ denote the post-softmax attention weight at layer ℓ, head index m (using m to avoid collision with the vocab token a of §3.2 and the RSP matrix $\mathbf{H}$), from query position i to key position j, satisfying $\sum_j \alpha_{i,j}^{(\ell,m)} = 1$ under the causal mask. Let Q and R denote the index sets of question and RSP tokens with sizes $|Q|$ and $|R| = L$, and let N_{head} be the number of heads. The per-token attention mass that query i directs to each group at layer ℓ is

$$\mathrm{PTM}_Q[\ell, i] = \frac{1}{|Q|\, N_{\text{head}}} \sum_{m=1}^{N_{\text{head}}} \sum_{j \in Q} \alpha_{i,j}^{(\ell,m)}, \qquad \mathrm{PTM}_R[\ell, i] = \frac{1}{|R|\, N_{\text{head}}} \sum_{m=1}^{N_{\text{head}}} \sum_{j \in R} \alpha_{i,j}^{(\ell,m)}. \quad (7)$$

Normalization by $|Q|$ and $|R|$ controls for group size, so both quantities express how much attention a single question (resp. RSP) token receives on average under the same softmax denominator. Question tokens thereby serve as a size-matched baseline that absorbs sequence-length effects common to both groups.

I.2 Heatmap Aggregation

We partition the layer indices and the reasoning position indices each into five equal-size bins $\{B_L^{(b)}\}_{b=1}^5$ and $\{B_P^{(b)}\}_{b=1}^5$. For sample s and target group $\bullet \in \{Q, R\}$, the value in cell (b_L, b_P) is

$$M_\bullet^{(s)}[b_L, b_P] = \frac{1}{|B_L^{(b_L)}| \cdot |B_P^{(b_P)}|} \sum_{\ell \in B_L^{(b_L)}} \sum_{i \in B_P^{(b_P)}} \mathrm{PTM}_\bullet[\ell, i]. \quad (8)$$

Figure 2 reports the sample average of Eq. (8) over the 500 valid samples.

I.3 Excluding the `\boxed{...}` Span

The reasoning position range terminates strictly before the final `\boxed{...}` span. Chain-of-thought outputs decompose into a *text* (content) component and a *pattern* (template) component

that contribute independently to downstream performance [Madaan and Yazdanbakhsh, 2022]. The `\boxed{...}` wrapper is a canonical pattern: a short, stereotyped span whose role is to complete the answer format rather than to extend reasoning. Including it would therefore inject a template-driven attention signature unrelated to reasoning dynamics.

I.4 Excluding the Final Two Layers

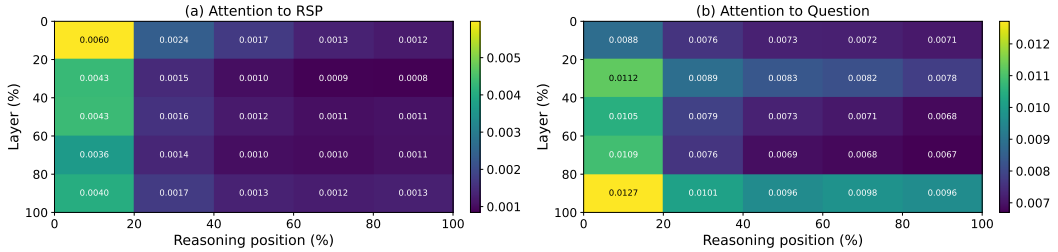
The layer axis further excludes the last two transformer layers. This choice is motivated by a standard late-layer effect rather than by any model-specific tuning decision.

Output-projection specialization. Late-layer hidden states lie in an approximately linear relationship with vocabulary logits and therefore function as a progressive linear readout rather than as representation-transforming blocks [Belrose et al., 2023]. Consistently, late-layer feed-forward modules have been characterized as output-distribution shaping mechanisms [Geva et al., 2021]. Attention in these layers therefore reflects output formation rather than reasoning computation.

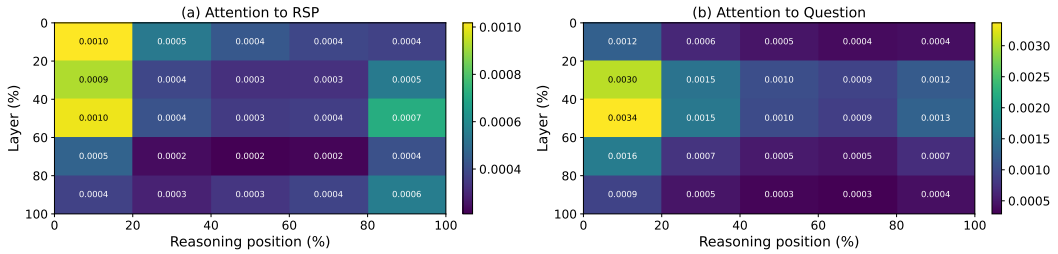
Excluding the final two layers keeps the heatmap focused on reasoning-phase dynamics instead of readout-phase dynamics. This exclusion is not load-bearing for the main claim: the sequence-direction attenuation emphasized in the main text is also visible without it, and Theorem 1 gives an exact per-head identity, with generation and layer trends assessed empirically.

I.5 Additional *Suffix* Heatmaps

For the late-layer reasons discussed above, the pattern in Figure 6 does not generalize uniformly across every cell. Nevertheless, the overall trend is consistent across both models: question-token attention varies broadly across layers, whereas RSP attention shows sequence-wise decay interpreted through the exact identity in Theorem 1, alongside an empirical layer-wise attenuation.



(a) Qwen2.5-Math-1.5B-Instruct, *suffix*



(b) LLaMA-3.1-8B-Instruct, *suffix*

Figure 6: Per-token RSP attention mass under *suffix* injection for the remaining two models (500 MATH-500 samples each). Axes and preprocessing match Figure 2: x-axis is reasoning position binned into five quantiles, y-axis is layer depth binned into five quantiles (top = shallow), color encodes the 500-sample mean per-token attention assigned to RSP tokens, and tokens inside `\boxed{}` spans and the last two layers are excluded.

J Additional DAPO Experiment

We additionally explore whether RSP can be combined with DAPO training [Yu et al., 2025], a variant of GRPO [Shao et al., 2024]. RSP is applied during rollout collection, while evaluation uses no RSP. Each condition uses a single training run.

J.1 From GRPO to DAPO

The vanilla GRPO objective [Shao et al., 2024] for a group of G rollouts $\{y_i\}_{i=1}^{G}$ (each $y_i = (y_{i,1}, \ldots, y_{i,|y_i|})$ a token sequence, and we use y rather than o throughout this appendix to avoid collision with the head output $\mathbf{o}_i^{(\ell)}$ of §3.2) with rewards $\{R_i\}_{i=1}^{G}$ is

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|y_i|}\sum_{t=1}^{|y_i|}\Big(\min\big(r_{i,t}\hat{A}_{i,t},\ \text{clip}(r_{i,t}, 1{-}\varepsilon, 1{+}\varepsilon)\hat{A}_{i,t}\big) - \beta\, D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}})\Big)\right], \tag{9}$$

with importance ratio $r_{i,t} = \pi_\theta(y_{i,t} \mid q, y_{i,<t}) / \pi_{\theta_{\text{old}}}(y_{i,t} \mid q, y_{i,<t})$, group-relative advantage $\hat{A}_{i,t} = (R_i - \text{mean}(\{R_i\}))/\text{std}(\{R_i\})$, clipping range ε, and KL penalty $\beta\, D_{\text{KL}}$ against the reference π_{ref}.

We build on the SimpleRL-Zoo [Zeng et al., 2025] training pipeline and implement DAPO on top of VeRL [Sheng et al., 2025]. DAPO modifies Eq. (9) in four ways: (i) token-level loss aggregation $1/\sum_i |y_i|$ instead of $1/|y_i|$, (ii) asymmetric clipping with $(\varepsilon_{\text{low}}, \varepsilon_{\text{high}}) = (0.2, 0.28)$, (iii) an implementation-side negative-advantage dual-clipping safeguard $\max(L_{\text{clipped}}, c\hat{A})$ with $c = 10$ (active only when $\hat{A} < 0$ and omitted from Eq. (10) below for readability), and (iv) KL terms removed. The resulting DAPO objective is

$$\mathcal{J}_{\text{DAPO}}(\theta) \;=\; \mathbb{E}\left[\frac{1}{\sum_i |y_i|}\sum_{i,t}\min\Big(r_{i,t}\hat{A}_{i,t},\ \text{clip}\big(r_{i,t},\ 1{-}\varepsilon_{\text{low}},\ 1{+}\varepsilon_{\text{high}}\big)\hat{A}_{i,t}\Big)\right]. \tag{10}$$

J.2 Training Setup and RSP Implementation

We implement DAPO on top of SimpleRL-Zoo [Zeng et al., 2025] and train Qwen2.5-Math-7B on a MATH Level 3–5 subset of approximately 8,500 prompts for 100 steps. Each step uses $G = 8$ rollouts per prompt at $\tau = 1.0$, a training batch size of $1{,}024$, and four PPO mini-batches of 256. We compare DAPO with DAPO + RSP, which uses independently sampled *suffix* RSPs with $L = 20$.

For DAPO + RSP, each rollout receives $\mathbf{H}_i \in \mathbb{R}^{L \times d}$. Each $\mathbf{H}_i$ is generated deterministically from a per-step rollout seed and injected via a forward hook. It is not a trainable parameter, so the learning signal is the policy adapting to arbitrary $\mathbf{H}_i$ rather than a learned prompt. Including $\mathbf{H}_i$ in the log-probabilities at both rollout and update time keeps the update on-policy with respect to the RSP-conditioned distribution (avoiding off-policy mismatch).

Table 20: DAPO / DAPO + RSP training setup on Qwen2.5-Math-7B.

Field	Value
Train data	MATH Level 3–5 subset (`simplelr_math_35`, ~8,500 prompts)
Prompt template	`qwen-boxed`
Train batch size	1024
PPO mini-batch size	256 (4 PPO updates per rollout)
Rollouts per prompt G	8
Max prompt / response length	2,028 / 2,048
Rollout sampling	temperature 1.0, top-p 1.0, top-k 50
$(\varepsilon_{\text{low}}, \varepsilon_{\text{high}})$	$(0.2,\ 0.28)$
Dual clip c	10.0
Loss aggregation	token-mean ($1/\sum_i \lvert y_i \rvert$)
KL terms	disabled
Entropy coefficient	0
Optimizer	AdamW, learning rate 5×10^{-7} (constant), no warmup
Total training	100 optimization steps
Save / eval frequency	every 10 steps
RSP (DAPO + RSP only)	*suffix* injection, $L = 20$, freshly drawn per rollout
Hardware	4× NVIDIA B200 GPUs
Wall-clock training time	∼12 hours per run

J.3 Evaluation Protocol

We evaluate on GSM8K [Cobbe et al., 2021], MATH-500 [Lightman et al., 2024], College Math, Minerva Math [Lewkowycz et al., 2022], and AIME24. AIME24 uses Avg@32 at $\tau = 1.0$, while the other four benchmarks use greedy decoding.

Each saved checkpoint is converted to HuggingFace format and evaluated using the SimpleRL-Zoo `eval_math_nodes.sh` pipeline, which selects sampling parameters per benchmark to reduce variance on the smaller competition sets:

Table 21: Per-benchmark evaluation protocol used by SimpleRL-Zoo `sh/eval.sh`.

Benchmark	# Problems	Temperature	n_{sampling}	Metric
AIME 2024	30	1.0	32	Avg@32
AMC 2023	40	1.0	32	Avg@32
MATH-500	500	0.0	1	accuracy (mean@1)
GSM8K	1,319	0.0	1	accuracy
OlympiadBench	675	0.0	1	accuracy
College Math	2,818	0.0	1	accuracy
Minerva Math	272	0.0	1	accuracy
GaoKao 2023-EN	385	0.0	1	accuracy

The maximum number of generated tokens is 16,000 across all benchmarks. The reported average is the unweighted mean over the five benchmarks (GSM8K, MATH-500, College Math, Minerva Math, AIME 2024).

J.4 Results

Table 22: Per-benchmark accuracy (%) at each method's peak step on Qwen2.5-Math-7B. **Bold** denotes the better of the two RL methods on each benchmark. Avg is the unweighted five-benchmark mean.

Method	GSM8K	MATH-500	College	Minerva	AIME24	Avg
Base	57.2	52.6	18.3	13.6	8.6	30.06
DAPO	86.3	78.2	41.4	37.5	22.6	53.20
DAPO + RSP	**87.7**	**78.6**	**42.2**	**39.7**	**23.6**	**54.36**

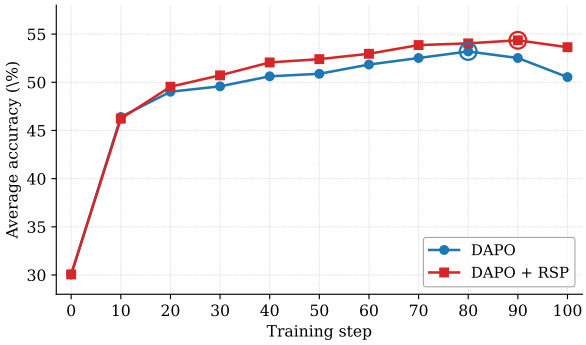


Figure 7: **DAPO training trajectories.** Five-benchmark average accuracy on Qwen2.5-Math-7B. Each condition uses a single training run.

Figure 7 and Table 22 summarize the observed training trajectories and each condition's peak checkpoint. DAPO + RSP reaches 54.36% at step 90, compared with DAPO's peak of 53.20% at step 80, a difference of 1.16 pp. At step 100, the difference is 3.10 pp. The curves cross around step 20.

Table 23: Five-benchmark average accuracy (%) at every checkpoint, computed over GSM8K, MATH-500, College Math, Minerva Math, and AIME 2024. **Bold** marks each method's peak.

Step	0	10	20	30	40	50	60	70	80	90	100
DAPO	30.06	46.40	49.02	49.58	50.62	50.88	51.84	52.52	**53.20**	52.52	50.54
DAPO + RSP	30.06	46.22	49.54	50.72	52.06	52.40	52.96	53.86	54.04	**54.36**	53.64

K Broader Impacts

This work provides empirical and theoretical analysis of how random embedding injection affects the reasoning behavior of large language models. The contributions are primarily foundational, with potential applications to training-data collection for reasoning models.

Positive impacts. RSP-induced trajectory diversity can help collect correct solutions for problems that the initial model solves infrequently. Our SFT experiments suggest that such data collection can improve subsequent model performance without requiring RSP at inference. This could make self-generated training data more useful for reasoning-oriented fine-tuning.

Potential concerns. Methods that expand the effective output support of LLMs may increase the reachability of low-probability tokens, which includes both desirable alternative reasoning paths and potentially unsafe or off-distribution outputs. Practitioners applying RSP in production systems should combine it with existing safety filtering and alignment mechanisms rather than treating it as a standalone diversity source.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction distinguish learned prompt content from embedding injection itself, summarize the theoretical and empirical analysis of RSP-induced diversity, and describe its application to training-data collection and subsequent SFT.

   (i) The per-layer RSP contribution decomposes exactly (Eq. (2)) into a real-token term and a contribution governed by a single attention-mass scalar $w_{r,i}^{(\ell)}$, which Theorem 1 expresses exactly through token counts and a generation-dependent log-sum-exp gap. Empirical attention and gap measurements support an implicit explore-then-exploit dynamic (§3.3).

   (ii) Proposition 1 shows that isotropic covariance maximizes worst-case directional variance under a fixed variance budget. Gaussian full support additionally gives positive probability to nonempty open branching regions. This concerns changes in token rankings, which temperature scaling preserves. Empirically, RSP at $\tau = 1.0$ produces Spearman $\rho = 0.709$ against the $\tau = 1.0$ baseline first-token distribution (vs $\rho = 1$ for any temperature) and retains 73.30% accuracy, compared with 1.42% at $\tau = 3.0$ (Table 6).

   (iii) RSP shows early-decoding distributional patterns related to those of optimized *soft prompts*, despite carrying no learned content (Figure 3). The controlled interventions in §4.3 further support a contribution from embedding injection itself. In the Pass@N experiments, independent-seed RSP at $\tau = 1.0$ reaches 92.80% on MATH-500 and 36.67% on AIME24 at $N = 16$, while a shared single-seed RSP matches the baseline on MATH-500 and improves AIME24 from 20.00% to 30.00%, remaining below independent resampling on both benchmarks (Figure 4).

   (iv) Section 4.6 evaluates rejection-sampling SFT with Qwen2.5-Math-7B-Instruct. Under the same rollout budget, RSP increases problem coverage from 23.75% to 49.68% (Table 8). Subsequent SFT improves five-benchmark average accuracy from 42.11% to 43.77%, with no RSP at evaluation (Table 9). The comparison includes differences in both the amount and composition of the collected data.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed alongside the relevant analyses. The Pass@N argument requires a positive probability of reaching a correct trajectory and independent rollouts. Isotropy and full support alone do not guarantee correctness (§3.3). The theoretical analysis uses a local first-order approximation for branch opening, and isotropy is a design criterion in input space rather than a property guaranteed throughout the transformer (§3.2).

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The paper contains four formal theoretical results, each with explicit assumptions and a proof. (i) Equation (2) (head-output decomposition into a renormalized real-token term and an RSP-induced contribution) carries a one-line in-text derivation in §3.2, using only the softmax-normalization identity under finite attention logits with $n \geq 1$ real and $L \geq 1$ RSP tokens. (ii) Theorem 1 (exact RSP attention mass) is proved inline in §3.3, assuming scaled dot-product attention with $n \geq 1$ real keys, $L \geq 1$ RSP keys, finite logits, and the token-count-normalized log-sum-exp gap g_i as defined in the statement. (iii) Proposition 1 (maximin directional coverage) is stated in §3.2 with its full proof in Appendix G, under the second-moment budget $\mathrm{tr}(\Sigma_D) \leq \rho^2$ on zero-mean distributions over $\mathbb{R}^d$. (iv) The Pass@N ensemble bound $\Pr(\text{Pass@}N \text{ on } x) \geq 1 - (1 - p_{\min}(x))^N$ is stated in §3.3 and proved formally in Appendix H.4.4, requiring (M1) the task-side condition

$p_{\min}(x) > 0$ and (M2) mutual independence of the N rollouts. The supporting top-K entry argument in §3.2, which combines full support of the RSP law with a locally affine surrogate $\Delta_{ab}(\bar{h}) \approx \Delta_{ab}(0) + b_{ab}^{\top}\bar{h}$ around $\bar{h} = 0$, is presented informally in the main text and written out formally in Appendix H.4.3.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The inference settings, RSP configurations, seed protocols, prompt templates, and answer-extraction procedures are provided in §4.1 and Appendix A. The mechanism analyses in §4.4–§4.5 use $L = 20$ with *suffix* injection, with sampling and metric settings specified in those sections and the corresponding appendices. For rejection-sampling SFT, §4.6 and Appendix A.4 describe hard-subset construction, rollout collection, correctness filtering, training hyperparameters, collection seeds, and evaluation without RSP.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The evaluation data and pretrained models used in the paper are publicly available. The RSP implementation is released at `https://github.com/heejunkim00/RSP` and linked from the project page (`https://heejunkim00.github.io/projects/RSP/`). Its README documents the environment and the run commands that reproduce the accuracy tables in §4.2, the Pass@N results in Figure 4, and the rejection-sampling SFT pipeline in §4.6, including data collection, grading, training, and evaluation.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The inference datasets and decoding settings are specified in §4.1. RSP lengths are reported in Appendix A, and the mechanism analyses in §4.4–§4.5 fix $L = 20$ with *suffix* injection. Appendix A.3 documents the training and optimization settings of TTSV, SoftCoT, and LTPO. For SFT, §4.6 defines the hard subset and the collection, training, and evaluation conditions. Appendix A.4 provides the learning rate, schedule, batch size, sequence length, seed settings, and benchmark-specific decoding protocols.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Tables 1 and 2 report mean accuracy and standard deviation over five seeds. The content probes report results over three seeds. Tables 8 and 9 report mean and standard deviation over three data-collection seeds, with the SFT seed fixed, as detailed in Appendix A.4. Figure 3 reports ± 1 SEM across problems. Table 6 reports accuracy variability across 16 samples per problem and averages token-level metrics over 16 RSP seeds. The corresponding captions and experimental descriptions specify the source of variability.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

Justification: The inference hardware is described in Appendix A. SFT uses eight NVIDIA H100 GPUs with 80GB memory each. Logged training runtimes per run are approximately 15 minutes for Baseline-SFT and 55 minutes for RSP-SFT, excluding data collection and evaluation (Appendix A.4). Decoding settings are specified separately for each experiment.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses publicly available math reasoning benchmarks and pretrained language models, involves no human subjects or personal data, and does not introduce a release that bypasses standard model-safety considerations. We have reviewed the NeurIPS Code of Ethics and identify no conflicts.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Appendix K discusses potential benefits from collecting correct solutions for problems with low initial solution success rates and improving subsequent SFT without RSP at inference. It also discusses the potential for greater trajectory diversity to increase the reachability of undesirable outputs and recommends combining RSP with existing safety filtering and alignment mechanisms.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: RSP itself is a training-free generation-time perturbation applied to existing pretrained models and adds no separate artifact requiring safeguards.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The datasets and pretrained models are credited through the references and source information provided in the paper. These include OpenR1-Math-46k [Yan et al., 2025] for SFT data collection, the evaluation benchmarks, and the Qwen2.5-Math [Yang et al., 2024] and LLaMA-3.1 [Grattafiori et al., 2024] models. The dataset card for the OpenR1-Math-46k distribution used in this work lists the MIT license, and its source link is provided in Appendix A.4. Each asset is used under its original publicly stated terms.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The RSP implementation (injection harness, evaluation scripts, and the SFT pipeline) is publicly released at `https://github.com/heejunkim00/RSP` under an open-source license, accompanied by a README that documents the environment, run commands, and the correspondence between scripts and the paper's tables and figures. The project page (`https://heejunkim00.github.io/projects/RSP/`) links the code and summarizes the main results.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper involves no crowdsourcing and no research with human subjects. All evaluations use static, publicly available benchmarks.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: As noted in item 14, the paper involves no human subjects, so IRB review is not applicable.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: Pretrained LLMs (Qwen2.5-Math, LLaMA-3.1) appear in the paper as the experimental subject of analysis, not as a non-standard component of the methods themselves. The methods (RSP definition and attention decomposition) do not rely on auxiliary LLMs in any non-standard or originality-affecting role.