EXPLORATION VIA ACTIVATION-SPACE STEERING FOR REINFORCEMENT LEARNING WITH VERIFIABLE REWARDS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning with verifiable rewards (RLVR) trains a language model by sampling responses, scoring them with an automatic verifier, and updating the policy from the resulting rewards. The policy therefore receives direct learning signals only from trajectories that appear in its rollouts. Yet repeated sampling from the same policy can concentrate on similar reasoning paths, leaving useful alternatives unexplored under a fixed rollout budget. We introduce Activation-Space Exploration (ASE), an online method for diversifying training rollouts. By clustering activation differences between correct and incorrect responses to the same problem, ASE builds a structured bank of latent exploration vectors. Different vectors are then applied to subsequent rollouts to explore alternative reasoning paths. On Qwen3-1.7B-Base and Qwen3-8B-Base trained on DAPO-Math-17k, ASE improves average Mean@8 across six mathematical reasoning benchmarks over GRPO by 4.87 and 1.87 percentage points, respectively, under the same rollout budget.

1 INTRODUCTION

Reinforcement learning improves a model by repeatedly sampling behaviors, evaluating their outcomes with rewards, and increasing the probability of behaviors that receive higher rewards (Sutton & Barto, 2018). Reinforcement learning with verifiable rewards (RLVR) makes this process scalable by using automated verifiers to assign outcome-based rewards to generated responses, such as mathematical solutions and programs (Le et al., 2022; Shao et al., 2024; Yu et al., 2025). A fundamental constraint, however, is that the policy can only learn from behaviors that appear in its sampled rollouts (Shao et al., 2024; Yue et al., 2025). Consequently, policy optimization is shaped by **which trajectories the rollout procedure makes available for learning**, a dependence that creates an exploration bottleneck under a finite rollout budget. Standard GRPO independently samples multiple trajectories from the same current policy (Shao et al., 2024). Increasing the number of rollouts provides more Monte Carlo samples, but leaves the underlying proposal distribution unchanged. As the policy concentrates probability mass on a limited set of reasoning patterns, multiple samples may repeatedly visit similar solution modes, reproduce the same intermediate steps, or make correlated errors. Useful alternatives that occupy low-probability regions of the policy can therefore remain rarely observed and receive little direct reward feedback. The bottleneck is thus not merely how many trajectories are sampled, but **how a fixed rollout budget is distributed across the policy's reasoning space**.

Several approaches seek to use a finite RLVR rollout budget more effectively. Adaptive sampling allocates more rollouts to difficult problems (Liao et al., 2025), while data filtering and curricula prioritize more informative training problems (Yu et al., 2025; Chen et al., 2025b). These methods change where sampling effort is spent, rather than how trajectories for the same problem are generated. Adjusting decoding temperature increases token-level variation (Dang et al., 2026), but does not specify which alternative reasoning patterns to explore. Replay reuses previously observed trajectories (Li et al., 2025), while novelty rewards and high-entropy prefix regeneration diversify generation around observed paths (Hu et al., 2026). Together, these methods improve how rollouts are allocated, change how randomly tokens are drawn, and reuse previously generated trajectories.

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

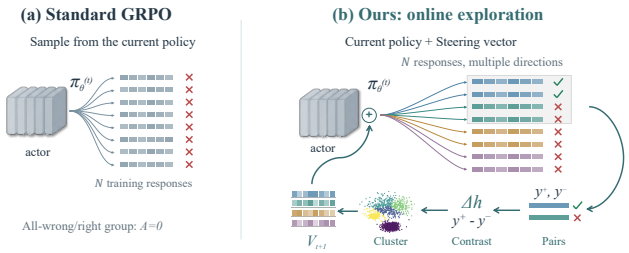

Figure 1: Motivation and high-level overview of activation-space exploration. Standard GRPO samples all responses from one proposal distribution, whereas ASE uses outcome feedback to maintain multiple activation-space directions, routes responses through different proposals, and updates the actor and vector bank online.

They leave the model's internal representations largely unexplored as a way to construct different rollout proposals for the same prompt.

We investigate activation-space exploration, where temporary interventions on hidden representations are used to construct multiple rollout proposal distributions while leaving both the problem context and model parameters unchanged. Activation steering has shown that directional interventions on Transformer representations can systematically alter model behavior and generation trajectories (Beaglehole et al., 2026; Chen et al., 2025a). Rather than using such interventions as inference-time behavior controllers, we ask whether they can serve as a training-time exploration mechanism: different latent directions can perturb generation toward different regions of the policy's reasoning space, exposing the optimizer to useful trajectories that ordinary sampling may rarely encounter.

Based on this idea, we propose **Activation-Space Exploration (ASE)**, an online exploration mechanism for RLVR that learns where to explore from the policy's own verifier results. For each problem, successful and unsuccessful trajectories generated during training provide within-problem contrasts that isolate activation differences associated with task outcome while controlling for problem content. We find that the activation contrasts exhibit heterogeneous structure; we therefore cluster and aggregate them into multiple exploration directions maintained in an online vector bank. During subsequent rollout generation, different trajectories for the same problem are routed through different latent directions, producing a collection of steering-conditioned proposal distributions rather than repeatedly sampling from a single one (Figure 1). Stochastic decoding is retained within every route, so ASE complements rather than replaces ordinary token sampling. Verifier rewards from the steered rollouts drive both subsequent exploration updates and optimization of the base policy, leaving evaluation and deployment intervention-free.

We evaluate ASE on Qwen3-1.7B-Base (Qwen Team, 2025) and Qwen3-8B-Base (Qwen Team, 2025) using DAPO-Math-17k (Yu et al., 2025) under the same rollout budget as standard GRPO. All activation interventions are disabled during evaluation. Across six mathematical reasoning benchmarks, ASE improves average Mean@8 over GRPO by **4.87 percentage points on Qwen3-1.7B-Base and 1.87 points on Qwen3-8B-Base**, while also transferring to out-of-distribution reasoning benchmarks. These results show that activation-space exploration during training can improve the learned policy across model scales and benchmark distributions under a fixed rollout budget.

2 Related Work

Reinforcement learning with verifiable rewards. RLVR is now a major paradigm for improving mathematical and code reasoning in language models. Shao et al. (2024) introduced GRPO, which constructs policy-gradient signals from relative rewards among multiple trajectories for the same problem; subsequent work studied the stability and efficiency of scaling this formulation (Yu et al., 2025). RLVR gains depend strongly on trajectories collected by the current policy. Policy optimization may mostly redistribute probability mass toward high-reward paths already available to the base model and can reduce solution-space coverage under large-scale sampling (Yue et al., 2025); policy entropy can also decrease continuously and eventually limit further exploration (Cui et al., 2025). At the token level, a small number of high-entropy branching positions have a disproportionate effect on reasoning paths (Wang et al., 2025). Recent methods encourage flatter distributions and answer diversity in the objective (Li et al., 2026), or combine trajectory novelty rewards with high-entropy

prefix regeneration to discover different correct solutions (Hu et al., 2026). RePO retrieves historical off-policy trajectories from a replay buffer and estimates their advantages separately from current on-policy rollouts (Li et al., 2025); GRESO uses reward dynamics across epochs to probabilistically skip prompts expected to remain zero-variance (Zheng et al., 2025b). These methods improve exploration through objectives, reward shaping, explicit trajectory operations, or data scheduling. We instead use historical rollout outcomes to construct multiple directional proposals in activation space, directly changing the sampling distribution of later training trajectories without an additional diversity reward or a modified GRPO objective.

Activation steering. Activation steering builds on the observation that high-level behaviors have controllable directions in language-model representations. Representation engineering and inference-time intervention show that identifying and modifying intermediate representations can change truthfulness, safety, and other behaviors (Zou et al., 2023; Li et al., 2023). More systematic studies have verified transferable and compositional linear concept representations across hundreds of concepts in language, vision-language, and reasoning models, and demonstrated their use for steering and monitoring (Beaglehole et al., 2026). Activation Addition (ActAdd) constructs a direction from activation differences of contrastive prompts (Turner et al., 2023); Contrastive Activation Addition (CAA) pairs examples with opposite behavior labels, averages residual-stream differences, and adds the result during generation (Rimsky et al., 2024). Later work extends this idea to instruction following (Stolfo et al., 2025) and reasoning behaviors such as uncertainty, backtracking, and hypothesis testing (Venhoff et al., 2025). Other methods automatically construct steering vectors from labeled data (Cui & Chen, 2026), or optimize a small number of trainable vectors with RL so that they carry reasoning gains (Sinii et al., 2026). These methods generally treat a steering vector as a behavior controller applied at inference time and rely on predefined attributes, externally constructed contrast sets, or optimization of the vector itself. We reposition activation steering as a temporary training-time exploration mechanism: directions are estimated online from success and failure trajectories of the current policy, several directions form heterogeneous rollout proposals, and GRPO absorbs useful behaviors into the base policy parameters. The vector bank is discarded after training.

3 METHOD

3.1 OVERVIEW

Let $x \sim \mathcal{D}$ be a problem and let π_θ be a language-model policy. For a fixed x, standard GRPO independently samples N trajectories:

$$y_i^{(t)} \sim \pi_{\theta^{(t)}}(\cdot \mid x), \qquad i = 1, \ldots, N. \tag{1}$$

With a fixed policy and sampling configuration, all trajectories are generated under the same proposal distribution. Increasing N gives more Monte Carlo samples while leaving the proposal distribution unchanged. We use historical rollout trajectories to construct structured activation-space exploration directions.

These exploration directions form a steering vector bank that is updated across iterations. At iteration t,

$$\mathcal{V}^{(t)} = \{\mathbf{v}_k^{(t)} \in \mathbb{R}^d\}_{k=1}^K, \qquad \|\mathbf{v}_k^{(t)}\|_2 = 1. \tag{2}$$

Here d is the hidden dimension of the target Transformer block and K is the number of exploration routes. Each vector is a smoothed direction obtained from mean activation differences between correct and incorrect trajectories in a cluster. The vector bank is a non-parametric training state independent of actor parameters and is estimated directly from historical rollouts.

During rollout, trajectories for the same problem are assigned different routes. For route $s_i \in \{1, \ldots, K\}$, we add $\alpha \mathbf{v}_{s_i}^{(t)}$ to the output of the target Transformer block during generation. The scalar α controls the intervention strength. We also include a clean route with no activation intervention, represented by $\mathbf{v}_0^{(t)} = \mathbf{0}$. This changes hidden states and subsequent token distributions, yielding a route-specific proposal distribution:

$$y_i^{(t)} \sim \begin{cases} \pi_{\theta^{(t)}}(\cdot \mid x), & s_i = 0, \\ \widetilde{\pi}_{\theta^{(t)}, \mathbf{v}_{s_i}^{(t)}}(\cdot \mid x), & s_i \in \{1, \ldots, K\}. \end{cases} \tag{3}$$

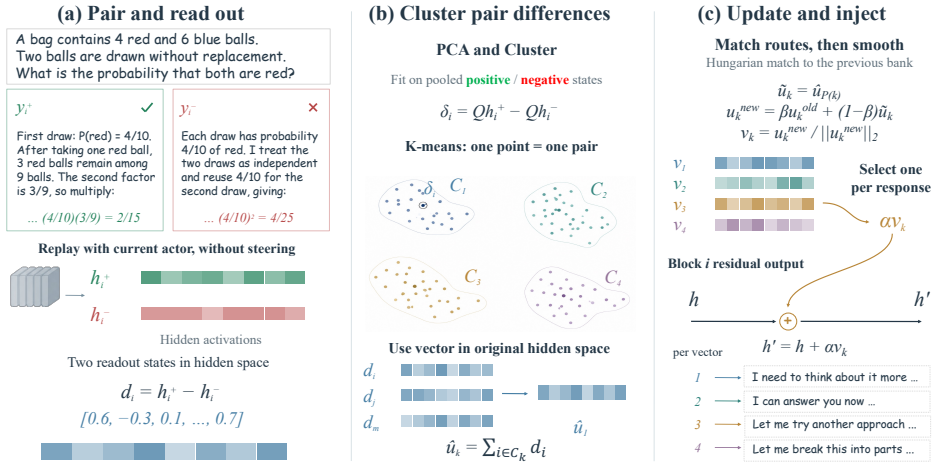

Figure 2: ASE pipeline. We replay successful and unsuccessful responses to read out hidden activations, cluster their pairwise differences to form exploration directions, match and smooth the directions across updates, and inject one selected direction into the residual stream for each routed response.

Token sampling remains stochastic within each route, and the main configuration allocates the group budget uniformly across the clean route and the K steering-guided proposals. If $\mathcal{V}^{(t)} = \emptyset$, the rollout distribution reduces to the unsteered policy $\pi_{\theta^{(t)}}(\cdot \mid x)$.

The verifier supplies a reward $r_i^{(t)}$ after generation. GRPO converts group rewards into relative advantages and updates the base policy with these routed trajectories. Steering changes only the proposal distribution during rollout, allowing training to discover candidates rarely covered by ordinary sampling. GRPO then increases the probability of useful behaviors under the base policy, making the exploratory experience persistent. Validation and deployment disable steering, so the resulting policy can reason independently.

The same verifier-scored trajectories update future exploration. We construct positive–negative pairs to estimate $\mathcal{V}^{(t+1)}$; a newly estimated bank becomes active only after actor optimization and weight synchronization. The resulting loop is structured exploration, outcome evaluation, policy learning, and exploration update (Figure 2). Section 3.2 describes vector-bank estimation and Section 3.3 describes steering-guided rollout optimization.

3.2 Online Exploration Vector Bank

Within-problem positive–negative pairing. For each problem, we compare trajectories with positive and negative rewards. Positive trajectories are successful samples from the current policy, while negative trajectories provide unsuccessful references. Let $\mathcal{P} = \{i : r_i = 1\}$ and $\mathcal{N} = \{i : r_i = 0\}$. Only groups containing both reward types provide a within-problem contrast. For each such group, we randomly select and pair $n_{\text{pair}} = \min(|\mathcal{P}|, |\mathcal{N}|)$ trajectories from the two sides. Groups whose rewards are all one or all zero do not contribute to the current steering state.

Activation readout. We replay each selected full prompt–response sequence with the current actor without steering. Let $\mathbf{h}_i = \phi_{\ell,p}(x, y_i) \in \mathbb{R}^d$ be the activation selected by position function p from the output of Transformer block ℓ. Each within-problem pair gives a raw activation difference $\mathbf{d}_j = \mathbf{h}_j^+ - \mathbf{h}_j^-$ for $j = 1, \ldots, n_{\text{pair}}$. Because the paired trajectories share the same prompt and use the same readout position rule p, this difference removes much of the variation caused by problem content and difficulty and provides a basis for structured exploration directions.

Paired activations accumulate across iterations in a buffer $\mathcal{B}$. Let $B_{\min}$ be the minimum number of pairs required for an update, Δ the minimum interval between updates, and t_{last} the iteration index of the most recent vector-bank update. When $|\mathcal{B}| \geq B_{\min}$ and $t - t_{\text{last}} \geq \Delta$, we re-estimate the vector bank from the full buffer and then clear it. The readout layer, position function, $B_{\min}$, and Δ are implementation hyperparameters.

Difference clustering. A single global mean difference would compress diverse exploration patterns into one direction. These patterns may reflect variation in early strategy choice, the sequencing of intermediate steps, or later checking and revision. We therefore cluster in a dimension-reduced pair-difference space.

Suppose the buffer contains M pairs $\{(\mathbf{h}_b^+, \mathbf{h}_b^-)\}_{b=1}^M$ collected across problems. To stabilize clustering when the number of pairs is limited relative to the hidden dimension, we fit a PCA projection $\mathbf{Q} \in \mathbb{R}^{d_{\mathrm{PCA}} \times d}$ on the pooled positive and negative activations and compute each pair's difference in this shared subspace as $\boldsymbol{\delta}_b = \mathbf{Q}\mathbf{h}_b^+ - \mathbf{Q}\mathbf{h}_b^-$. Here d_{PCA} denotes the number of retained principal components. K-means partitions $\{\boldsymbol{\delta}_b\}_{b=1}^M$ into K clusters $\{\mathcal{C}_k^{(t+1)}\}_{k=1}^K$. The set $\mathcal{C}_k^{(t+1)}$ contains the indices of all pairs assigned to cluster k. PCA only defines the clustering space. The final direction is the within-cluster mean difference in the original hidden space:

$$\widehat{\mathbf{u}}_k^{(t+1)} = \frac{1}{|\mathcal{C}_k^{(t+1)}|} \sum_{b \in \mathcal{C}_k^{(t+1)}} (\mathbf{h}_b^+ - \mathbf{h}_b^-). \tag{4}$$

The clusters construct multiple local rollout proposals; we do not assume that they correspond to discrete, stable, or human-namable semantics.

Cross-iteration stabilization. Let $\mathbf{u}_k^{(t)}$ denote the unnormalized, temporally smoothed direction, with $\mathbf{v}_k^{(t)} = \mathbf{u}_k^{(t)}/\|\mathbf{u}_k^{(t)}\|_2$. A finite buffer is affected by rollout noise, so we use an EMA with coefficient $\beta \in [0, 1)$ to smooth updates. Since K-means cluster indices have no fixed meaning across updates, we first align the new candidate directions with the existing bank by solving

$$P^{(t+1)} = \arg\max_{P \in \mathfrak{S}_K} \sum_{k=1}^{K} \cos(\mathbf{u}_k^{(t)}, \widehat{\mathbf{u}}_{P(k)}^{(t+1)}), \tag{5}$$

where $\mathfrak{S}_K$ is the set of permutations. We use the Hungarian algorithm, then update

$$\mathbf{u}_k^{(t+1)} = \beta \mathbf{u}_k^{(t)} + (1-\beta)\widehat{\mathbf{u}}_{P^{(t+1)}(k)}^{(t+1)}. \tag{6}$$

For the first update, we set $\mathbf{u}_k^{(t+1)} = \widehat{\mathbf{u}}_k^{(t+1)}$. Finally, we normalize $\mathbf{u}_k^{(t+1)}$ to unit L_2 norm to obtain $\mathbf{v}_k^{(t+1)}$. The new bank is staged until actor optimization and rollout weight synchronization finish.

3.3 Steering-Guided Rollout Optimization

Routed activation steering. When $\mathcal{V}^{(t)} \neq \emptyset$, we assign routes $s_i \in \{0, \ldots, K\}$ to trajectories for each problem, where $s_i = 0$ is the clean route. Our main method uses all $K+1$ routes uniformly within each group. For any target-layer output $\mathbf{h}^{(\ell)}$ on a trajectory assigned to a steering route $s_i \in \{1, \ldots, K\}$, the intervention is

$$\widetilde{\mathbf{h}}^{(\ell)} = \mathbf{h}^{(\ell)} + \alpha \mathbf{v}_{s_i}^{(t)}. \tag{7}$$

For the clean route, the output is unchanged. The intervention is applied to every forward pass of the target layer, including prompt prefill and autoregressive decoding. Since each vector has unit L_2 norm, α directly controls the injected residual. Steering is executed by the rollout engine with fixed actor parameters and is outside backpropagation.

GRPO update. For the N rollouts of a fixed problem x, let μ and σ be the mean and sample standard deviation of their verifier rewards. The group-relative advantage is $A_i = (r_i - \mu)/(\sigma + \epsilon)$, where $\epsilon > 0$ stabilizes normalization. We update the base policy using trajectories from both clean and steered rollout routes, and apply a bounded token-level importance weight in the clipped GRPO objective. For each response token,

$$\rho_{i,\tau}(\theta) = \frac{\pi_\theta(y_{i,\tau} \mid x, y_{i,<\tau})}{\pi_{\theta^{(t)}}(y_{i,\tau} \mid x, y_{i,<\tau})}. \tag{8}$$

For $s_i \in \{0, \ldots, K\}$, the truncated importance ratio is

$$w_{i,\tau} = \min\left(c_{\mathrm{IS}}, \frac{\pi_{\theta^{(t)}}(y_{i,\tau} \mid x, y_{i,<\tau})}{\widetilde{\pi}_{\theta^{(t)}, \mathbf{v}_{s_i}^{(t)}}(y_{i,\tau} \mid x, y_{i,<\tau})}\right). \tag{9}$$

Here $c_{\mathrm{IS}} > 0$ caps the importance weight. This weight is computed separately at each response position. The token-level objective is

$$\mathcal{L}(\theta) = -\mathbb{E}_{x,i,\tau}[w_{i,\tau} \min\left(\rho_{i,\tau}(\theta)A_i, \mathrm{clip}(\rho_{i,\tau}(\theta), 1-\varepsilon, 1+\varepsilon)A_i\right)]\,. \tag{10}$$

The expectation averages over generated response tokens, and $\varepsilon > 0$ sets the PPO clipping range. Minimizing this objective updates the base policy from $\theta^{(t)}$ to $\theta^{(t+1)}$. The verifier and advantage estimator follow GRPO. The bounded importance weight adjusts the contribution of each sampled token according to its likelihood under the base policy and its rollout route. The complete training procedure is given in Algorithm 1.

4 Experiments

4.1 Experimental Setup

Models and datasets. We train Qwen3-1.7B-Base and Qwen3-8B-Base (Qwen Team, 2025) with DAPO-Math-17k (Yu et al., 2025). We report six mathematical test benchmarks: MATH500 (Hendrycks et al., 2021; Lightman et al., 2023), Minerva (Lewkowycz et al., 2022), AIME24 (Zhang & Math-AI Team, 2024), AIME25 (Zhang & Math-AI Team, 2025), AMC24 (ExtraordinaryLab, 2025), and OlympiadBench (He et al., 2024); and three out-of-distribution benchmarks: MMLU-Pro (Wang et al., 2024), GPQA-Diamond (Rein et al., 2024), and BBH (Suzgun et al., 2023).

Baselines. We compare with standard GRPO and four representative baselines. DAPO (Yu et al., 2025) dynamically identifies and discards zero-variance prompts; Dr.GRPO (Liu et al., 2025) redesigns advantages and loss aggregation to address length bias; GSPO (Zheng et al., 2025a) performs importance weighting, clipping, and optimization at the sequence level; and RePO (Li et al., 2025) reuses complete historical trajectories from a replay buffer. We follow method-specific configurations from the corresponding papers or public implementations.

Implementation details. All methods use four epochs, a training and actor mini-batch size of 64, and eight rollouts per problem. The actor is optimized with AdamW (learning rate 1×10^{-6}, weight decay 0.01), a constant learning-rate schedule, and no warmup; we use neither KL loss nor entropy regularization. Training rollouts use temperature 1.0, top-p 1.0, top-$k = -1$, and maximum prompt/response lengths of 1024/4096, while final evaluation uses temperature 0.6, top-p 0.95, top-$k = 0$, and maximum response length 4096. For ASE, we use $K = 3$ steering clusters plus a clean route, steering strength $\alpha = 0.5$, and EMA coefficient $\beta = 0.5$. Training uses verl (Sheng et al., 2025), vLLM (Kwon et al., 2023), BF16, and FlashAttention-2 (Dao, 2024) on four NVIDIA A800 GPUs for the 1.7B model and eight for the 8B model. Math-Verify[1] checks final answers. Shared training and evaluation settings are given in Appendix C; checkpoint-selection and remaining method-specific details are provided in Appendices B and C.

4.2 Main Results

Tables 1 and 2 present the main results for Qwen3-8B-Base and Qwen3-1.7B-Base, respectively. Each table cell reports Mean@8 / Pass@8 in percent. Mean@8 is the average accuracy over eight sampled answers; Pass@8 is the fraction of problems with at least one correct answer. Reported scores are averaged over three runs. Math Avg. and OOD Avg. are the unweighted means of benchmark-level scores across the six mathematical and three out-of-distribution benchmarks, respectively, computed separately for Mean@8 and Pass@8.

ASE-GRPO achieves the highest mathematical Mean@8 average on both model sizes. On Qwen3-1.7B-Base, it reaches 30.34, compared with 25.47 for GRPO and 27.73 for Dr.GRPO; its mathematical Pass@8 also leads at 48.85. On Qwen3-8B-Base, it reaches 50.83, exceeding GRPO by 1.87 points and GSPO by 1.89 points. Improvements are particularly clear on AMC24 and OlympiadBench, while other methods remain stronger on Minerva or AIME24.

[1] https://github.com/huggingface/Math-Verify

Table 1: Main results for Qwen3-8B-Base trained on DAPO-Math-17k. Scores are reported as Mean@8 / Pass@8 (in percent). Reported scores are averaged over three runs. Each Δ row reports the absolute difference from the corresponding baseline in percentage points.

	Mathematical benchmarks							OOD benchmarks			
Method	MATH500	Minerva	AIME24	AIME25	AMC24	Olympiad	Math Avg.	MMLU-Pro	GPQA-D	BBH	OOD Avg.
GRPO	88.17/94.87	**43.98**/55.76	26.39/48.89	21.39/**37.78**	58.06/82.22	55.80/70.18	48.96/64.95	**67.62**/85.64	49.45/79.80	76.73/91.19	64.60/85.54
DAPO	86.09/93.20	42.56/54.17	21.53/41.11	18.06/30.00	48.98/74.07	51.38/66.82	44.77/59.89	61.28/83.38	46.70/80.81	76.71/91.32	61.56/85.17
Dr.GRPO	87.83/**95.87**	43.20/**59.44**	25.42/47.78	20.69/**37.78**	55.56/77.78	57.78/**74.78**	48.41/**65.57**	65.20/**86.00**	48.34/**84.85**	**77.79**/91.99	63.78/**87.61**
GSPO	88.05/95.00	43.01/55.51	25.00/**53.33**	**22.50**/36.67	58.33/80.00	56.71/72.40	48.94/65.49	64.94/84.08	47.47/80.81	75.74/91.20	62.72/85.36
RePO	86.09/94.53	41.57/54.29	22.64/42.22	18.61/32.22	53.06/81.48	53.34/71.36	45.89/62.69	61.21/84.40	43.96/81.31	67.92/91.35	57.69/85.69
ASE-GRPO	**89.31**/95.73	43.70/56.74	**28.19**/46.67	21.67/36.67	**62.78/82.96**	**59.32**/74.09	**50.83**/65.48	66.63/85.68	**51.01**/82.32	76.26/**92.74**	**64.63**/86.91
Δ vs. GRPO	+1.14/+0.86	-0.28/+0.98	+1.80/-2.22	+0.28/-1.11	+4.72/+0.74	+3.52/+3.91	+1.87/+0.53	-0.99/+0.04	+1.56/+2.52	-0.47/+1.55	+0.03/+1.37
Δ vs. DAPO	+3.22/+2.53	+1.14/+2.57	+6.66/+5.56	+3.61/+6.67	+13.80/+8.89	+7.94/+7.27	+6.06/+5.59	+5.35/+2.30	+4.31/+1.51	-0.45/+1.42	+3.07/+1.74
Δ vs. Dr.GRPO	+1.48/-0.14	+0.50/-2.70	+2.77/-1.11	+0.98/-1.11	+7.22/+5.18	+1.54/-0.69	+2.42/-0.09	+1.43/-0.32	+2.67/-2.53	-1.53/+0.75	+0.85/-0.70
Δ vs. GSPO	+1.26/+0.73	+0.69/+1.23	+3.19/-6.66	-0.83/+0.00	+4.45/+2.96	+2.61/+1.69	+1.89/-0.01	+1.69/+1.60	+3.54/+1.51	+0.52/+1.54	+1.91/+1.55
Δ vs. RePO	+3.22/+1.20	+2.13/+2.45	+5.55/+4.45	+3.06/+4.45	+9.72/+1.48	+5.98/+2.73	+4.94/+2.79	+5.42/+1.28	+7.05/+1.01	+8.34/+1.39	+6.94/+1.22

Table 2: Main results for Qwen3-1.7B-Base trained on DAPO-Math-17k. Scores are reported as Mean@8 / Pass@8 (in percent). Reported scores are averaged over three runs. Each Δ row reports the absolute difference from the corresponding baseline in percentage points.

	Mathematical benchmarks							OOD benchmarks			
Method	MATH500	Minerva	AIME24	AIME25	AMC24	Olympiad	Math Avg.	MMLU-Pro	GPQA-D	BBH	OOD Avg.
GRPO	65.59/83.47	27.37/43.50	6.39/14.44	5.42/**22.22**	18.24/40.74	29.82/50.59	25.47/42.50	35.46/69.09	26.45/79.12	44.70/84.15	35.54/77.45
DAPO	68.02/84.20	27.94/45.22	10.00/**26.67**	5.00/18.89	21.20/43.70	30.87/50.49	27.17/44.86	39.22/69.27	29.12/76.60	50.43/85.28	39.59/77.05
Dr.GRPO	68.16/84.53	**30.39**/45.59	**12.08**/23.33	2.92/12.22	21.39/40.74	31.43/51.68	27.73/43.02	40.54/70.33	**30.30**/77.61	51.99/85.43	40.94/77.79
GSPO	68.42/84.80	28.71/44.36	9.58/22.22	4.86/18.89	18.24/39.26	32.00/53.81	26.97/43.89	38.38/70.15	27.90/78.96	47.02/84.75	37.77/77.95
RePO	65.42/84.67	26.19/44.61	7.08/22.22	4.31/15.56	17.78/42.22	29.60/51.88	25.06/43.53	36.15/69.45	26.54/76.26	44.36/84.96	35.68/76.89
ASE-GRPO	**71.15/88.40**	29.78/**47.06**	10.83/25.56	**8.19**/21.11	**27.96/56.30**	**34.12/54.65**	**30.34/48.85**	**41.35/71.12**	29.95/**79.97**	**52.46/85.87**	**41.25/78.98**
Δ vs. GRPO	+5.56/+4.93	+2.41/+3.56	+4.44/+11.12	+2.77/-1.11	+9.72/+15.56	+4.30/+4.06	+4.87/+6.35	+5.89/+2.03	+3.50/+0.85	+7.76/+1.72	+5.71/+1.53
Δ vs. DAPO	+3.13/+4.20	+1.84/+1.84	+0.83/-1.11	+3.19/+2.22	+6.76/+12.60	+3.25/+4.16	+3.17/+3.99	+2.13/+1.85	+0.83/+3.37	+2.03/+0.59	+1.66/+1.93
Δ vs. Dr.GRPO	+2.99/+3.87	-0.61/+1.47	-1.25/+2.23	+5.27/+8.89	+6.57/+15.56	+2.69/+2.97	+2.61/+5.83	+0.81/+0.79	-0.35/+2.36	+0.47/+0.44	+0.31/+1.19
Δ vs. GSPO	+2.73/+3.60	+1.07/+2.70	+1.25/+3.34	+3.33/+2.22	+9.72/+17.04	+2.12/+0.84	+3.37/+4.96	+2.97/+0.97	+2.05/+1.01	+5.44/+1.12	+3.48/+1.03
Δ vs. RePO	+5.73/+3.73	+3.59/+2.45	+3.75/+3.34	+3.88/+5.55	+10.18/+14.08	+4.52/+2.77	+5.28/+5.32	+5.20/+1.67	+3.41/+3.71	+8.10/+0.91	+5.57/+2.09

The gains extend beyond the mathematical training domain. ASE-GRPO obtains the highest OOD Mean@8 average on both Qwen3-1.7B-Base (41.25) and Qwen3-8B-Base (64.63), compared with 35.54 and 64.60 for GRPO, respectively. The 1.7B model also achieves the highest OOD Pass@8 average, whereas Dr.GRPO remains higher on this metric for the 8B model. Together, these results suggest that activation-space exploration improves the unsteered policy across model scales, with consistent gains in mathematical reasoning and evidence of transfer to out-of-distribution tasks.

Figure 3 extends the evaluation to larger sampling budgets. ASE-GRPO achieves higher math-benchmark Pass@k than GRPO across $k = 8$ to 128 on both model scales. This suggests that the learned policy is more effective at discovering correct solutions when sampling multiple candidate responses.

4.3 Dynamics of Training and Vector-Bank

Figure 4 tracks validation performance and rollout behavior on Qwen3-1.7B-Base and Qwen3-8B-Base. Validation Mean@8 rises during training, and ASE-GRPO reaches higher values than GRPO on both model scales. Training reward follows a similar trajectory for the two methods, and response lengths remain within a comparable range. We use pairwise disagreement between answers sampled for the same problem as an answer-level measure of exploration. The disagreement gap (ASE-GRPO minus GRPO) is positive overall and generally grows over training on both model sizes, showing that ASE-GRPO explores a broader range of answers alongside its validation gains without a marked change in training reward or response length.

Figure 5 shows how ASE-GRPO's vector bank evolves on Qwen3-1.7B-Base. Each of the three cluster shares is the fraction of activation pairs assigned to that cluster at a bank update. The shares remain broadly stable and reasonably balanced throughout training, showing that all three directions continue to receive support from the contrast data. In the *Vector geometry* panel, temporal alignment is the mean cosine similarity between newly extracted vectors and their matched vectors from the previous bank; the median cosine of 0.73 shows that successive vectors remain similarly oriented. Inter-cluster alignment is the mean pairwise cosine similarity among directions in the updated bank; values near the three-vector lower bound of -0.5 show that the directions remain well separated.

4.4 Inference-Time Steering versus Training-Time Steering

Activation steering can be used in two distinct ways. An inference-time intervention changes the output distribution of a fixed model during evaluation, whereas our training-time intervention di-

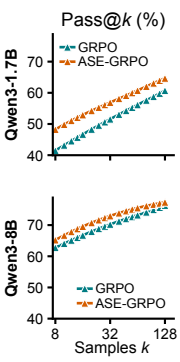

Figure 3: Pass@k averaged over six math benchmarks (in percent).

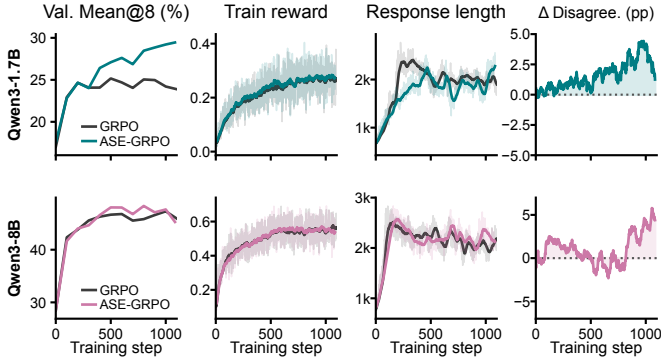

Figure 4: Training dynamics of GRPO and ASE-GRPO on Qwen3-1.7B-Base and Qwen3-8B-Base. The four columns show validation Mean@8, training reward, response length, and the rollout disagreement gap across training steps.

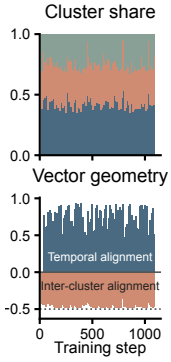

Figure 5: Vector-bank dynamics for Qwen3-1.7B-Base.

Table 3: Training- versus inference-time steering on Qwen3-1.7B-Base. ^{S} denotes inference-time steering. Math/OOD Avg. report Mean@8 / Pass@8 (%), averaged over three evaluations.

Condition	Math Avg.	OOD Avg.
Base	18.74/39.31	18.02/61.16
Base^{S}	18.86/40.09	17.93/61.51
GRPO	25.47/42.50	35.54/77.45
GRPO^{S}	25.81/43.13	35.83/77.29
ASE-GRPO	30.34/48.85	41.25/78.98
ASE-GRPO^{S}	30.85/49.09	41.40/78.44

Table 4: Direction-construction ablation on Qwen3-1.7B-Base. ASE-GRPO variants differ only in direction construction. Math/OOD Avg. report Mean@8 / Pass@8 (%), averaged over three evaluations.

Condition	Math Avg.	OOD Avg.
GRPO	25.47/42.50	35.54/77.45
Random directions	27.76/45.62	39.48/78.09
Shuffled labels	27.56/44.46	39.93/77.03
Reversed directions	27.87/45.15	39.13/78.48
ASE-GRPO	30.34/48.85	41.25/78.98

versifies the rollouts used to train the policy. We compare these two uses of steering to determine whether the main benefit comes from the direct effect of steering a fixed policy or from cumulative gains of online exploration during training. We compare six conditions: Base, Base^{S}, GRPO, GRPO^{S}, ASE-GRPO, and ASE-GRPO^{S}, where ^{S} denotes inference-time intervention. To assess the potential gains from inference-time steering, we sweep its parameters separately for each checkpoint; details are reported in Appendix D.

Table 3 summarizes the comparison. Inference-time intervention provides only marginal changes for the untrained and GRPO checkpoints: Base^{S} changes Math Mean@8 from 18.74 to 18.86 and GRPO^{S} changes it from 25.47 to 25.81. ASE-GRPO already outperforms standard GRPO without inference-time intervention, and applying inference-time intervention to ASE-GRPO gives a further 0.51-point Math Mean@8 increase and 0.15-point OOD Mean@8 increase, while OOD Pass@8 decreases from 78.98 to 78.44. These results indicate that the main gain comes from training-time exploration, with only modest direct effects from steering a fixed policy.

4.5 Ablating Verifier-Guided Direction

The vector bank is estimated from within-problem success–failure contrasts. This raises a specific attribution question: does verifier feedback matter, or is the gain explained by injecting any additional activation perturbation? We therefore design a strictly matched ablation in which only the construction of the exploration directions varies. Random directions replace each cluster candidate with an independent random vector of the same norm; shuffled labels randomly swap the positive and negative labels within each pair before clustering and direction estimation; and reversed directions negate the learned vectors while preserving their cluster assignments. These controls retain the same downstream bank update and routing procedure.

As shown in Table 4, ASE-GRPO outperforms all three ablation variants on both aggregates, reaching 30.34/48.85 on Math Avg. and 41.25/78.98 on OOD Avg. The variants mostly exceed GRPO (25.47/42.50 and 35.54/77.45, respectively), except that shuffled labels has lower OOD Pass@8 (77.03 versus 77.45). These variants retain the clustered $K = 3$ bank, routing, and EMA updates, which may contribute to their gains over GRPO. An additional control without clustering or EMA yields only a small increase in Math Mean@8 and a decrease in Pass@8 relative to GRPO (Appendix D.2), suggesting that arbitrary random steering alone does not provide a consistent benefit.

Table 5: ASE-Dr.GRPO and ASE-GSPO on Qwen3-1.7B-Base. Entries are Mean@8 / Pass@8 (%); Math Avg. and OOD Avg. cover six mathematical and three OOD benchmarks, respectively.

	Mathematical benchmarks							OOD benchmarks			
Method	MATH500	Minerva	AIME24	AIME25	AMC24	Olympiad	Math Avg.	MMLU-Pro	GPQA-D	BBH	OOD Avg.
Dr.GRPO	68.27/84.60	**31.25**/47.06	**12.08**/20.00	2.92/**16.67**	20.83/40.00	31.36/**52.67**	27.79/43.50	40.36/**70.62**	29.86/77.27	**52.03/85.61**	40.75/**77.83**
ASE-Dr.GRPO	**69.40/86.20**	30.06/**47.43**	11.25/**26.67**	**5.83/16.67**	**24.17/44.44**	**33.70/52.67**	**29.07/45.68**	**40.45**/69.23	**31.06/77.78**	51.81/84.95	**41.11**/77.32
Δ vs. Dr.GRPO	+1.13/+1.60	-1.19/+0.37	-0.83/+6.67	+2.91/+0.00	+3.34/+4.44	+2.34/+0.00	+1.28/+2.18	+0.09/-1.39	+1.20/+0.51	-0.22/-0.66	+0.36/-0.51
GSPO	69.00/86.20	28.72/43.75	8.75/**23.33**	5.00/**23.33**	19.17/42.22	**32.64**/54.30	27.21/45.52	38.15/70.48	26.96/**80.81**	46.91/84.55	37.34/78.61
ASE-GSPO	**69.80/87.00**	**30.19/47.43**	**10.00/23.33**	**6.25**/13.33	**23.33/48.89**	32.60/**54.75**	**28.70/45.79**	**40.48/72.52**	**28.98**/78.79	**48.80/85.53**	**39.42/78.95**
Δ vs. GSPO	+0.80/+0.80	+1.47/+3.68	+1.25/+0.00	+1.25/-10.00	+4.16/+6.67	-0.04/+0.45	+1.49/+0.27	+2.33/+2.04	+2.02/-2.02	+1.89/+0.98	+2.08/+0.34

Table 6: Training-time overhead on matched GRPO and ASE-GRPO runs. Throughput uses all prompt and response tokens, rollout latency is normalized by output tokens, and actor-update latency is normalized by all valid tokens.

Model	Method	Step time (s/step) ↓	Training throughput (k tokens/s/GPU) ↑	Rollout latency (μs/output token) ↓	Actor-update time (μs/valid token) ↓
1.7B	GRPO	92.21	2.845	52.91	29.15
1.7B	ASE-GRPO	93.05	2.666	54.35	33.60
1.7B	Δ vs. GRPO	+0.91%	−6.29%	+2.72%	+15.27%
8B	GRPO	136.10	1.041	60.08	49.35
8B	ASE-GRPO	145.88	0.974	61.59	56.08
8B	Δ vs. GRPO	+7.19%	−6.44%	+2.51%	+13.64%

Together, these results suggest that the shared bank structure and EMA may account for part of the improvement, while verifier-guided directions provide additional gains within the matched setup.

4.6 Evaluation on Alternative Policy Optimization Formulations

To test whether the benefit of ASE is specific to the standard GRPO formulation, we apply the same training-time exploration mechanism to Dr.GRPO and GSPO. We use Qwen3-1.7B-Base, the same four-epoch training budget, and eight rollouts per problem for all conditions. We do not perform a separate parameter search for either alternative formulation and use the preset ASE configuration. Table 5 summarizes the results. Compared with Dr.GRPO, ASE-Dr.GRPO increases Math Avg. Mean@8 and Pass@8 by 1.28 and 2.18 points, respectively, and OOD Avg. Mean@8 by 0.36 points. Compared with GSPO, ASE-GSPO increases Math Avg. Mean@8 and Pass@8 by 1.49 and 0.27 points, respectively, and OOD Avg. Mean@8 and Pass@8 by 2.08 and 0.34 points, respectively. These results indicate that activation-space exploration also benefits alternative policy-optimization formulations.

4.7 Training-time overhead.

Table 6 compares latency and throughput for the matched GRPO and ASE-GRPO runs, averaged over the training trajectory after excluding the first 10 warm-up steps. ASE-GRPO increases token-normalized rollout latency by only 2.72% on the 1.7B model and 2.51% on the 8B model, while end-to-end throughput decreases by 6.29% and 6.44%, respectively. The additional activation replay and vector-bank maintenance are reflected in the actor-update latency, but the resulting mean step-time increases remain modest at 0.91% and 7.19%. Moreover, the vector bank is discarded after training, so deployment incurs no additional inference latency.

5 Conclusion

We introduced Activation-Space Exploration (ASE), an online method that uses verifier feedback to construct and update multiple activation-space directions for training rollouts. By distributing a fixed rollout budget across these directions, ASE improves mathematical reasoning on two model scales and also benefits Dr.GRPO and GSPO. Matched ablations show that success–failure contrasts yield stronger gains than random, label-shuffled, or reversed directions, while comparisons with inference-time steering support the value of using these interventions during training. All steering is removed at evaluation, so the gains persist in the learned base policy without additional inference-time intervention. These findings support activation space as a complementary exploration axis for RLVR.

AI USE STATEMENT

In this work, we used generative AI tools to design and provide feedback on research methodology and experiments, assist with the execution of experiments, and assist with translation of parts of the manuscript. We also used generative AI tools for sourcing and searching for research information, identifying relevant literature, creating and editing software code used in the experiments, and formatting references. We did not use generative AI tools to generate synthetic datasets, develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for mathematical proofs, assist in writing proofs, propose or refine hypotheses, clean or reformat datasets, support qualitative or thematic data analysis, or interpret the reported results. All AI-assisted work was reviewed by the authors. Experimental procedures, software code, outputs, citations, translations, and reported claims were checked and verified by the authors. We take responsibility for the final content of this work, including all text, claims, results, and artifacts produced with the aid of generative AI.

REFERENCES

Daniel Beaglehole, Adityanarayanan Radhakrishnan, Enric Boix-Adserà, and Mikhail Belkin. Toward universal steering and monitoring of AI models. *Science*, 391(6787):787–792, 2026. doi: 10.1126/science.aea6792. URL https://doi.org/10.1126/science.aea6792.

Runjin Chen, Andy Arditi, Henry Sleight, Owain Evans, and Jack Lindsey. Persona vectors: Monitoring and controlling character traits in language models. *arXiv preprint arXiv:2507.21509*, 2025a. URL https://arxiv.org/abs/2507.21509.

Xiaoyin Chen, Jiarui Lu, Minsu Kim, Dinghuai Zhang, Jian Tang, Alexandre Piché, Nicolas Gontier, Yoshua Bengio, and Ehsan Kamalloo. Self-evolving curriculum for LLM reasoning, 2025b. URL https://arxiv.org/abs/2505.14970.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, and Ning Ding. The entropy mechanism of reinforcement learning for reasoning language models, 2025. URL https://arxiv.org/abs/2505.22617.

Sasha Cui and Zhongren Chen. Painless activation steering: An automated, lightweight approach for post-training large language models, 2026. URL https://arxiv.org/abs/2509.22739.

Haoran Dang, Cuiling Lan, Hai Wan, Xibin Zhao, and Yan Lu. Temperature as a meta-policy: Adaptive temperature in LLM reinforcement learning. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=AoTHU2OmS6.

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations*, 2024.

ExtraordinaryLab. AMC dataset, 2025. URL https://huggingface.co/datasets/extraordinarylab/amc.

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 *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, 2024. doi: 10.18653/v1/2024.acl-long.211. URL https://doi.org/10.18653/v1/2024.acl-long.211.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Wenyang Hu, Junxiang Jia, Zhen Shu, Daniel Dahlmeier, See-Kiong Ng, and Bryan Kian Hsiang Low. ExTra: Exploratory trajectory optimization for language model reinforcement learning, 2026. URL https://arxiv.org/abs/2606.24994.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven Chu Hong Hoi. CodeRL: Mastering code generation through pretrained models and deep reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 21314–21328, 2022. doi: 10.52202/068431-1549. URL https://doi.org/10.52202/068431-1549.

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 *Advances in Neural Information Processing Systems*, volume 35, pp. 3843–3857, 2022. doi: 10.52202/068431-0278. URL https://doi.org/10.52202/068431-0278.

Changhao Li, Yuchen Zhuang, Chenxiao Gao, Haotian Sun, Rushi Qiang, Chao Zhang, and Bo Dai. Exploration-driven optimization for test-time large language model reasoning. *Transactions on Machine Learning Research*, May 2026. URL https://openreview.net/forum?id=NiINDlzvNj.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In *Advances in Neural Information Processing Systems*, volume 36, pp. 41451–41530. Curran Associates, Inc., 2023. doi: 10.52202/075280-1797. URL https://doi.org/10.52202/075280-1797.

Siheng Li, Zhanhui Zhou, Wai Lam, Chao Yang, and Chaochao Lu. RePO: Replay-enhanced policy optimization, 2025. URL https://arxiv.org/abs/2506.09340.

Mengqi Liao, Xiangyu Xi, Ruinian Chen, Jia Leng, Yangen Hu, Ke Zeng, Shuai Liu, and Huaiyu Wan. Enhancing efficiency and exploration in reinforcement learning for LLMs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 1451–1463, Suzhou, China, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.75. URL https://aclanthology.org/2025.emnlp-main.75/.

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

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding R1-Zero-like training: A critical perspective. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=5PAF7PAY2Y.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof Q&A benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15504–15522, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.828. URL https://doi.org/10.18653/v1/2024.acl-long.828.

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, 2024. URL https://arxiv.org/abs/2402.03300.

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 *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, 2025. doi: 10.1145/3689031.3696075. URL https://doi.org/10.1145/3689031.3696075.

Viacheslav Sinii, Nikita Balagansky, Gleb Gerasimov, Daniil Laptev, Yaroslav Aksenov, Vadim Kurochkin, Alexey Gorbatovski, Boris Shaposhnikov, and Daniil Gavrilov. Small vectors, big effects: A mechanistic study of RL-induced reasoning via steering vectors, 2026. URL https://arxiv.org/abs/2509.06608.

Alessandro Stolfo, Vidhisha Balachandran, Safoora Yousefi, Eric Horvitz, and Besmira Nushi. Improving instruction-following in language models through activation steering. In *International Conference on Learning Representations*, pp. 55790–55823, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/8c3262a4c965ba9888f120d4f9e13478-Abstract-Conference.html.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018. URL http://incompleteideas.net/book/the-book-2nd.html.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. Challenging BIG-Bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, 2023. doi: 10.18653/v1/2023.findings-acl.824. URL https://doi.org/10.18653/v1/2023.findings-acl.824.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering, 2023. URL https://arxiv.org/abs/2308.10248.

Constantin Venhoff, Iván Arcuschin, Philip Torr, Arthur Conmy, and Neel Nanda. Understanding reasoning in thinking language models via steering vectors. In *ICLR 2025 Workshop on Reasoning and Planning for Large Language Models*, 2025. URL https://iclr.cc/virtual/2025/32763.

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 *Advances in Neural Information Processing Systems*, volume 38, pp. 115452–115486. Curran Associates, Inc., 2025. doi: 10.52202/085713-3850. URL https://doi.org/10.52202/085713-3850.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems*, volume 37, pp. 95266–95290, 2024. doi: 10.52202/079017-3018. URL https://doi.org/10.52202/079017-3018.

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 *Advances in Neural Information Processing Systems*, volume 38, pp. 113222–113244. Curran Associates, Inc., 2025. doi: 10.52202/085713-3775. URL https://doi.org/10.52202/085713-3775.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In *Advances in Neural Information Processing Systems*, volume 38, pp. 57654–57689. Curran Associates, Inc., 2025. doi: 10.52202/085713-1933. URL https://doi.org/10.52202/085713-1933.

Yifan Zhang and Math-AI Team. American invitational mathematics examination (AIME) 2024, 2024. URL https://huggingface.co/datasets/math-ai/aime24.

Yifan Zhang and Math-AI Team. American invitational mathematics examination (AIME) 2025, 2025. URL https://huggingface.co/datasets/math-ai/aime25.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization, 2025a. URL https://arxiv.org/abs/2507.18071.

Haizhong Zheng, Yang Zhou, Brian Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. Act only when it pays: Efficient reinforcement learning for LLM reasoning via selective rollouts. In *Advances in Neural Information Processing Systems*, volume 38, pp. 124321–124346. Curran Associates, Inc., 2025b. doi: 10.52202/085713-4143. URL https://doi.org/10.52202/085713-4143.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to AI transparency, 2023. URL https://arxiv.org/abs/2310.01405.

A Complete Training Algorithm

Algorithm 1 summarizes the implementation of ASE-GRPO.

Algorithm 1 ASE-GRPO.

Input: initial policy $\pi_{\theta^{(0)}}$, training distribution $\mathcal{D}$, rollout count N, bank size K, steering strength α, EMA coefficient β, minimum pair count $B_{\min}$, and update interval Δ.

Initialize: $\mathcal{V}^{(0)} \leftarrow \emptyset$ and paired-activation buffer $\mathcal{B} \leftarrow \emptyset$.

1. For $t = 0, 1, 2, \ldots$, sample a batch of problems from $\mathcal{D}$ and generate routed rollouts with $\pi_{\theta^{(t)}}$ and $\mathcal{V}^{(t)}$.
2. Compute binary verifier rewards, then balance and pair correct and incorrect trajectories within each group.
3. Replay the selected trajectories with the unsteered actor, read the corresponding activations, and append the paired activations to $\mathcal{B}$.
4. If $|\mathcal{B}| \geq B_{\min}$ and at least Δ iterations have elapsed, fit PCA on the pooled activations and run K-means on the PCA-space pair differences. Compute cluster means in the original hidden space, match candidates to historical directions, apply EMA and normalization, stage $\mathcal{V}^{(t+1)}$, and clear $\mathcal{B}$.
5. Compute group-relative advantages and update the base policy with the clipped GRPO objective.
6. Synchronize actor weights to the rollout engine. Activate a staged vector bank if one exists; otherwise set $\mathcal{V}^{(t+1)} \leftarrow \mathcal{V}^{(t)}$.

Output: the base policy; discard the vector bank.

B Evaluation and Checkpoint Selection

Training uses only the local DAPO-Math-17k training file. The original JSONL contains 17,398 samples; filtering by maximum prompt length retains 17,390 samples in the standard configuration, while RePO uses 17,344 samples under its fixed rollout configuration.

We use MATH500 for test-set adaptive model selection by evaluating it periodically during training. For ASE and all baselines, we select the checkpoint with the highest MATH500 Mean@8. We

evaluate MATH500 every 100 optimization steps using eight responses per problem, temperature 0.6, top-p 0.95, top-$k = 0$, and seed 42. Evaluation on Minerva, AIME24, AIME25, AMC24, OlympiadBench, and the out-of-distribution benchmarks (MMLU-Pro, GPQA-Diamond, and Big-Bench Hard) is performed only on the selected checkpoint for each experiment.

C Implementation Details

C.1 Common training and evaluation settings

Table 7: Common settings for the two model sizes.

Configuration	Qwen3-1.7B-Base	Qwen3-8B-Base
Training data	DAPO-Math-17k	DAPO-Math-17k
Training epochs	4	4
Train/actor mini-batch	64/64	64/64
Rollouts per problem	8	8
Max prompt/response length	1024/4096	1024/4096
Optimizer	AdamW	AdamW
Learning rate / weight decay	1×10^{-6}/0.01	1×10^{-6}/0.01
LR scheduler / warmup	constant / 0	constant / 0
Train sampling	1.0 / 1.0 / -1	1.0 / 1.0 / -1
Validation/test sampling	0.6 / 0.95 / 0	0.6 / 0.95 / 0
Thinking mode	enabled	enabled
Training hardware	4 NVIDIA A800 GPUs	8 NVIDIA A800 GPUs

All training uses verl (Sheng et al., 2025), vLLM (Kwon et al., 2023) rollouts, BF16, and FlashAttention-2 (Dao, 2024). We filter samples exceeding the maximum prompt length. Except for method-specific objectives or reward processing, we use neither KL loss nor entropy regularization.

C.2 Baseline and ASE configurations

Table 8: Method-specific configurations.

Method	PPO clipping	Loss aggregation / policy loss	Extra processing
GRPO	(0.2, 0.2)	token-mean / vanilla GRPO	none
DAPO	(0.2, 0.28)	token-mean / vanilla GRPO	dynamic sampling; overlong buffer
Dr.GRPO	(0.2, 0.2)	seq-mean-token-sum-norm / vanilla GRPO	response normalization
GSPO	$(3 \times 10^{-4}, 4 \times 10^{-4})$	seq-mean-token-mean / GSPO	sequence-level loss
RePO	(0.2, 0.2)	seq-mean-token-mean / vanilla GRPO	replay cache 16; samples 8
ASE-GRPO	(0.2, 0.2)	token-mean / vanilla GRPO	routed activation steering

Table 9: ASE-GRPO configuration.

Configuration	Value
Intervention layer	Transformer block 0
Vector construction	CAA-style clustered positive-minus-negative differences
Number of routes K	3
Route allocation	balanced uniform
Clean response bank	enabled
Readout position function p	response token at 25% of response length
Steering strength α	0.5
EMA coefficient β	0.5
Paired buffer minimum $B_{\min}$	512 trajectory pairs
Vector-bank update interval Δ	10 training iterations
PCA dimension / solver	64 / randomized
K-means `n_init`	512

This subsection supplements the common settings above with method-specific configurations for the baselines and ASE-GRPO. The two tables summarize the policy optimization settings and extra processing used by each method, together with the activation readout, routing, steering, and vector-bank

update settings used by ASE-GRPO. For rollout correction, we use verl's default token-level truncated importance-sampling configuration, with $c_{\mathrm{IS}} = 2.0$ and no batch normalization of the weights. For layer choice, we read out activations and apply steering at the output of Transformer block 0 in all ASE experiments. We use a single intervention layer to avoid accumulating direct vector additions across multiple depths, while an early intervention allows the injected direction to propagate through the remaining blocks. This choice is heuristic: intermediate or later layers may provide more informative representations for constructing exploration directions. We did not run a layer sweep within the available compute budget, so we do not claim that block 0 is optimal; comparing readout and intervention layers, including multi-layer configurations, remains future work.

D Additional Ablation

D.1 Hyperparameter Sensitivity

We report ASE sensitivity sweeps for the number of clusters K, the EMA coefficient β, the steering strength α, and the response-relative activation readout position p. Each run uses its best validation checkpoint. Entries are Mean@8 / Pass@8 percentages; the mathematical average covers the six mathematical benchmarks and the OOD average covers MMLU-Pro, GPQA-D, and BBH. The sweeps vary one parameter at a time, with the remaining parameters fixed as indicated in each table.

Table 10: Sensitivity to the number of clusters K. For this ASE sweep, we fix $\alpha = 0.4$ and $\beta = 0.5$.

K	Math Avg.	OOD Avg.
2	29.60/45.74	40.08/78.44
3	29.64/47.02	39.62/76.52
4	27.01/44.63	39.40/78.24
5	28.04/46.38	38.82/78.26
6	28.04/44.54	39.54/77.72

Table 11: Sensitivity to the EMA coefficient β. For this ASE sweep, we fix $K = 3$ and $\alpha = 0.2$.

β	Math Avg.	OOD Avg.
0.0	26.48/42.63	35.84/72.25
0.1	28.29/45.41	38.02/72.10
0.3	26.27/46.10	35.33/76.78
0.5	30.21/46.44	40.79/79.71
0.7	29.41/48.83	37.14/79.14
0.9	29.35/47.81	39.89/78.35

Table 12: Sensitivity to the steering strength α. For this ASE sweep, we fix $K = 3$ and $\beta = 0.5$.

α	Math Avg.	OOD Avg.
0.2	30.21/46.44	40.79/79.71
0.3	28.38/48.05	38.75/77.62
0.4	29.64/47.02	39.62/76.52
0.5	30.31/48.67	41.35/79.97
0.6	29.21/44.63	39.43/76.84

The three sweeps suggest that a small route bank is sufficient under the tested rollout budget. For the cluster-count sweep, $K = 3$ gives the highest mathematical Mean@8 and Pass@8, at 29.64 and 47.02, respectively, whereas $K = 2$ gives the highest OOD Mean@8 and Pass@8, at 40.08 and 78.44. Increasing K beyond three does not produce a monotonic improvement: both mathematical metrics are lower for $K \in \{4, 5, 6\}$ than for $K = 3$. This indicates that adding routes alone does not guarantee better coverage, and that the available rollout evidence must also support reliable estimation of each direction.

The EMA sweep supports using temporal smoothing when updating the vector bank. Relative to the unsmoothed setting $\beta = 0$, the intermediate setting $\beta = 0.5$ improves mathematical Mean@8 from

26.48 to 30.21 and OOD Mean@8 from 35.84 to 40.79. It also gives the highest OOD Pass@8, 79.71, while the highest mathematical Pass@8 is obtained at $\beta = 0.7$, with 48.83. Thus, moderate-to-strong smoothing performs well across the reported metrics, with $\beta = 0.5$ providing a useful overall operating point rather than being the strict optimum for every metric.

The strength sweep shows non-monotonic variation across the tested values of α. Mathematical Mean@8 ranges from 28.38 to 30.31 and Pass@8 from 44.63 to 48.67; OOD Mean@8 ranges from 38.75 to 41.35 and Pass@8 from 76.52 to 79.97. All four metrics reach their highest values at $\alpha = 0.5$: 30.31/48.67 for Math Avg. and 41.35/79.97 for OOD Avg. Increasing the strength to $\alpha = 0.6$ lowers all four metrics, so stronger steering does not consistently improve the final unsteered policy over this range.

These tables are separate one-parameter sensitivity analyses rather than a joint factorial study. In particular, the cluster-count sweep fixes $\alpha = 0.4$, the EMA sweep fixes $\alpha = 0.2$, and the strength sweep varies α with $K = 3$ and $\beta = 0.5$. The different fixed strengths reflect separate experimental runs that inherited the working configuration used at the time of each sweep, rather than a rerun of every sweep after selecting a single global anchor. This design keeps each table's within-sweep comparison well-defined, but it prevents comparisons of absolute values across tables from being interpreted as a unified hyperparameter ranking. In particular, the best row in each table should not be combined to claim a jointly optimal (K, β, α) configuration. A fully matched sensitivity study would hold one common anchor fixed for all three scans, or would evaluate a prespecified joint grid; the present results are intended to characterize local sensitivity around the configurations used in the respective runs.

We also vary the response-relative readout position on Qwen3-1.7B-Base in a separate sweep, selecting activations at a fraction f of each response's length. Its absolute scores are not directly comparable with those of the preceding ablations since the sweeps were not conducted under a common matched configuration. Within this sweep, $f = 0.25$ performs best on the mathematical benchmarks, the primary target of ASE. One possible explanation is that earlier response states reflect the initial solution strategy before later calculations and answer formatting add trajectory-specific variation, making success–failure contrasts more reusable across problems. This explanation remains a hypothesis based on single-run comparisons.

Table 13: Sensitivity to the response-relative activation readout position. We fix $K = 3$, $\alpha = 0.2$, and $\beta = 0.7$; scores are Mean@8 / Pass@8 (%). Math Avg. and OOD Avg. use the same benchmark groups as the main tables.

Readout fraction f	Math Avg.	OOD Avg.
0.25	28.93/46.79	38.24/77.59
0.50	27.89/43.94	39.68/76.98
0.75	27.59/44.95	40.00/78.36

D.2 Ablating Verifier-Guided Direction

The matched controls in Table 4 change direction construction while retaining the clustered $K = 3$ bank, routing, and EMA updates. For random directions, each independent random vector is rescaled to the corresponding cluster candidate's original L_2 norm before the EMA update. For shuffled labels, the positive and negative labels within each activation pair are randomly exchanged before PCA, K-means, and cluster-direction estimation. Consequently, the controls' gains over GRPO do not isolate the effect of arbitrary activation perturbations: the shared bank structure and temporal averaging may also contribute.

To examine this possibility, we additionally use unstructured random steering: eight sampled unit random vectors, one per rollout, with no clustering or EMA. The vectors do not depend on activations or verifier labels. As shown in Table 14, this control reaches 26.65/41.59 on Math Avg., compared with 26.00/43.91 for GRPO and 30.31/48.67 for ASE-GRPO. Its Mean@8 is only 0.65 points above GRPO, while its Pass@8 is 2.32 points lower, so arbitrary random steering does not yield a consistent improvement in this comparison. This pattern is consistent with the structured bank and EMA contributing to the gains of the matched controls.

Table 14: Six-benchmark Math Avg. (Mean@8 / Pass@8, %). GRPO and ASE-GRPO use the first evaluation at their selected checkpoints.

Method	Math Avg.
GRPO	26.00/43.91
Unstructured random steering	26.65/41.59
ASE-GRPO	30.31/48.67

D.3 Inference-time steering sweep

The sweep compares three checkpoints: Qwen3-1.7B-Base and its GRPO- and ASE-GRPO-trained variants. For each model, vectors are constructed from response-relative activations captured on the 17,390 filtered DAPO-Math-17k prompts. We use block 0, within-problem verifier-reward pairing, CAA-style positive-minus-negative differences, no cluster-feature scaling, PCA dimension 64, K-means `n_init`=512, balanced-uniform route allocation, and a clean route. We evaluate

$$K \in \{1, 2, 3, 4, 5, 6, 7\}, \qquad f \in \{0.25, 0.50, 0.75\}, \qquad \alpha \in \{0.25, 0.50, 0.75, 1.00\}. \tag{11}$$

This gives 84 steering conditions per model. Each steering condition is independently repeated three times, with eight responses per problem. The Base and GRPO clean references use three standalone evaluations; the ASE-GRPO clean reference combines its selected-checkpoint training validation with two standalone evaluations. Evaluation uses MATH500, Minerva, AIME24, AIME25, and AMC24, temperature 0.6, top-p 0.95, top-$k = 0$, maximum response length 4096, and thinking mode.

For a sweep cell, Core Math Mean@8 and Pass@8 are the unweighted averages across the five benchmarks. We report the mean and sample standard deviation over the three evaluation repeats. The corresponding clean-relative changes use the same model's clean mean:

$$\Delta_{\text{mean}} = \overline{\text{Mean@8}}_{\text{steer}} - \overline{\text{Mean@8}}_{\text{clean}}, \qquad \Delta_{\text{pass}} = \overline{\text{Pass@8}}_{\text{steer}} - \overline{\text{Pass@8}}_{\text{clean}}. \tag{12}$$

Table 3 uses the highest Core Math Mean@8 configuration from each model's sweep: $(K, f, \alpha) = (7, 0.50, 0.50)$ for Base, $(7, 0.25, 0.25)$ for GRPO, and $(4, 0.75, 0.50)$ for ASE-GRPO. These sweep-selected results are interpreted as exploratory upper envelopes.

Table 15: Inference-time sweep grid and reporting units.

Quantity	Values or definition
Models	Base; standard GRPO; ASE-GRPO
Number of routes K	$1, 2, 3, 4, 5, 6, 7$
Capture fraction f	$0.25, 0.50, 0.75$
Strength α	$0.25, 0.50, 0.75, 1.00$
Conditions per model	$7 \times 3 \times 4 = 84$
Repeats	Three independent evaluations per cell
Responses	Eight per problem
Core Math Avg.	Unweighted average over five benchmarks
Selected Base (K, f, α)	$(7, 0.50, 0.50)$
Selected GRPO (K, f, α)	$(7, 0.25, 0.25)$
Selected ASE-GRPO (K, f, α)	$(4, 0.75, 0.50)$