LEARNING REACHABLE PLANS: PLANGAP-GUIDED DISTILLATION AND PLAN-CONTRASTIVE POLICY OPTIMIZATION FOR REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning with verifiable rewards improves long chain-of-thought (Long-CoT) reasoning in large language models (LLMs). However, outcome-supervised Group Relative Policy Optimization (GRPO) derives a single response-level advantage from each rollout's terminal reward and applies it across all tokens. To support fine-grinded advantages, we introduce a *Plan-and-Action* structure that places a concise high-level plan before detailed execution. We first validate this structure through a *PlanGap-guided distillation*, which extracts plans from verified teacher solutions and selects those with smaller teacher–student gaps in mean plan negative log-likelihood to prioritize strategies accessible to the student. We then propose Plan-Contrastive Group Relative Policy Optimization (**PC-GRPO**), which operates independently of the distillation stage and requires no SFT cold start. Under a fixed verifier budget, PC-GRPO samples multiple plans and multiple executions per plan, using their outcomes to estimate plan reliability and construct separate learning signals for plan selection and plan-conditional execution. It further contrasts plans with higher and lower estimated values using only plan tokens, providing plan-specific supervision without privileged solutions, an external teacher, or additional full-trajectory scoring for the contrastive signal. Experiments across diverse reasoning domains, model families, and scales show improved reasoning performance and more efficient plan supervision and policy optimization.

1 INTRODUCTION

Large language models (LLMs) use chain-of-thought (CoT) reasoning to develop intermediate steps before producing an answer (Wei et al., 2022). Within these trajectories, choosing a solution strategy and carrying it out poses different learning problems. A model may consistently follow an unsuitable plan or select a promising plan and fail due to a local execution error. Conversely, execution may revise an initially weak plan before reaching a correct answer. The final outcome alone does not reveal which of these events occurred. Longer trajectories offer more room for exploration and correction but also make it harder to attribute success or failure to particular strategic choices. Learning effective reasoning therefore requires evidence that distinguishes the quality of a plan from the outcome of an individual execution.

Explicit planning provides a useful interface for studying this distinction. Plan-then-action approaches expose a high-level plan before detailed reasoning, and planning-oriented post-training incorporates such structure into supervision and optimization (Parmar et al., 2025; Dou et al., 2026). Yet introducing a plan does not by itself establish how that plan should be learned. Prompted guidance depends on the quality of the proposed strategy, while supervised fine-tuning must determine which teacher-generated plans offer useful supervision for the student. Prior work shows that reasoning traces from stronger teachers need not match a smaller student's learning capacity (Li et al., 2025; Yan et al., 2026); post-training can also alter response length without corresponding accuracy gains (Wu et al., 2026). A plan extracted from a verified solution consequently provides a candidate learning target, but its source alone does not establish its suitability for the student. The question for plan distillation is therefore: ***which extracted plans should receive explicit supervision?***

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.

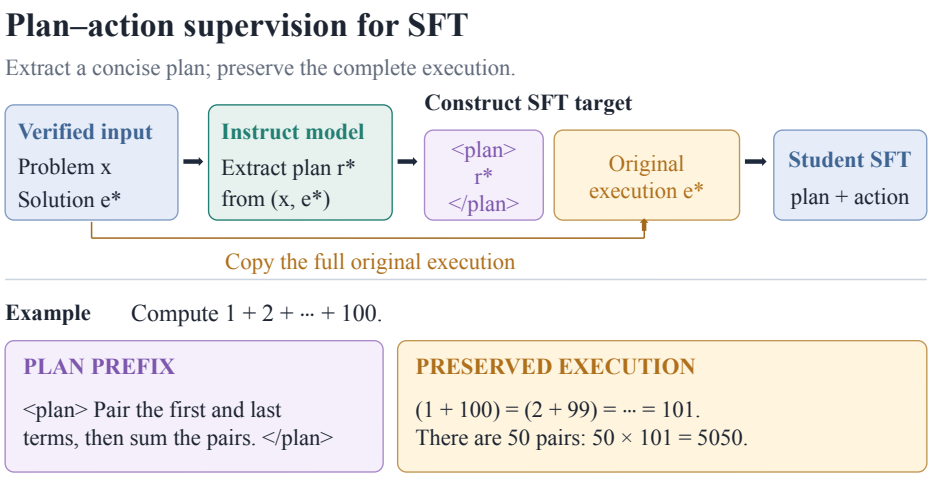

Figure 1: **PlanGap-guided distillation.** Concise plans are extracted from verified solutions while preserving the original executions. The teacher–student mean plan NLL gap guides selective plan augmentation; matched allocation experiments evaluate its utility.

Online learning presents a related attribution problem. Reinforcement learning with verifiable rewards grounds updates in answer correctness, but outcome-supervised Group Relative Policy Optimization (GRPO) assigns the same group-relative advantage to every token in a response (Shao et al., 2024). Planning and execution therefore receive the same outcome-derived signal despite their different roles. PTA-GRPO evaluates plans through repeated continuations and incorporates an analytic plan reward into a composite response reward (Dou et al., 2026). This supplies evidence about plan quality, while leaving room to distinguish plan-level learning from execution-specific feedback and to refine credit within the plan itself. Dense feedback offers another direction: on-policy self-distillation scores student trajectories using privileged solution context (Zhao et al., 2026), and RLCSD contrasts correct and incorrect solution contexts to modulate token-level credit (Pan et al., 2026). These approaches motivate a complementary training question: ***can repeated outcomes and alternative plans provide fine-grained strategic feedback while confining additional scoring to the plan span?***

We address both questions through a compact plan interface. As illustrated in Figure 1, *PlanGap-guided distillation* extracts plans from verified solutions and allocates plan supervision according to the teacher–student gap in mean plan negative log-likelihood. Low-gap examples receive plan prefixes, while the remaining examples retain their original executions. A matched study isolates this allocation decision. Our online method uses the same interface without an SFT cold start.

We propose **Plan-Contrastive Group Relative Policy Optimization (PC-GRPO)**, illustrated in Figure 2. For each problem, PC-GRPO samples multiple plans and executions. Mean execution success estimates plan reliability, while differences among executions of the same plan distinguish successful from failed attempts. Separate advantages supervise the corresponding token spans, allowing a useful plan to receive positive feedback even when an individual execution fails. PC-GRPO further contrasts sibling plans ranked by repeated-execution success. A frozen policy snapshot supplies plan-token scores, while bounded modulation preserves the direction determined by verified outcomes. Additional scoring covers only plan tokens and requires no reference solution or external teacher during RL. Inference uses one plan and one execution.

In summary, our contributions are as follows:

- We introduce *PlanGap-guided Plan Distillation*, which preserves verified executions and allocates concise plan supervision using a teacher–student reachability gap.
- We propose **PC-GRPO**, which uses repeated executions to distinguish plan reliability from execution quality and refines plan-token advantages through contrasts between sampled plans. It operates under a fixed verifier budget without requiring an SFT cold start or dense scoring of full trajectories.

- Across model families and scales, PC-GRPO improves matched GRPO while reducing measured training time. Mathematics-only training further transfers to science and coding benchmarks.

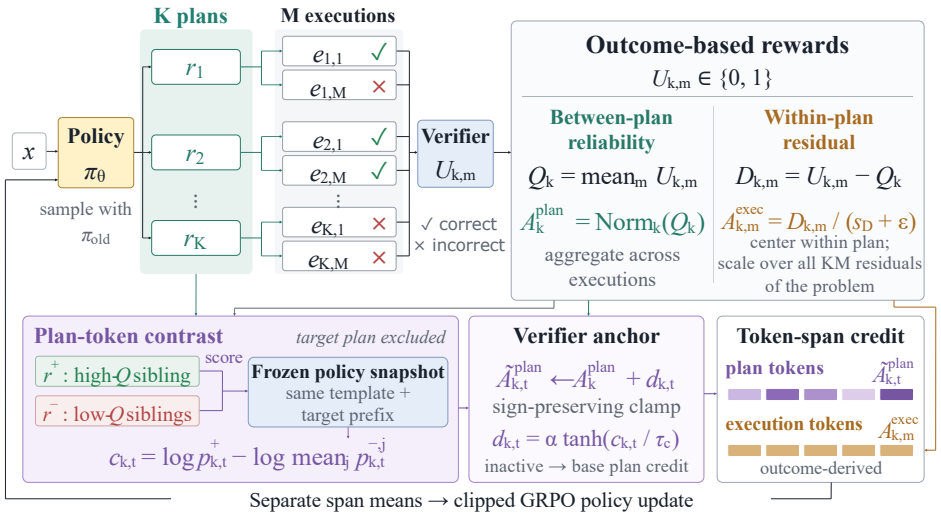

Figure 2: **Plan-Contrastive Group Relative Policy Optimization.** Repeated executions estimate plan reliability and conditional execution quality under a fixed verifier budget. Contrasts between alternative plans refine plan-token credit while preserving the verifier-derived advantage sign.

2 METHOD

We study selective plan distillation and PC-GRPO independently. The former allocates supervision from verified solutions, and the latter learns from plan-conditioned rollouts without an SFT cold start.

2.1 PROBLEM SETUP

Given a problem x, the policy generates a plan c and an execution y: $\pi_\theta(c, y \mid x) = \pi_\theta(c \mid x)\pi_\theta(y \mid x, c)$. Plan-then-execute reasoning follows prior planning interfaces (Wei et al., 2022; Wang et al., 2023a; Dou et al., 2026) and execution may revise the initial plan (Yao et al., 2023b). The `<plan>` delimiters identify disjoint plan and execution positions $\mathcal{T}_c$ and $\mathcal{T}_y$. An external verifier returns $R(x, y) \in \{0, 1\}$ (Shao et al., 2024; DeepSeek-AI, 2025).

Plan reliability under the rollout policy is $q_{\text{old}}(x, c) = \mathbb{E}_{y \sim \pi_{\text{old}}(\cdot \mid x, c)}[R(x, y)]$. PC-GRPO estimates this quantity through repeated executions, using differences across plans to assess plan reliability and differences within a plan to assess execution outcomes.

2.2 PLANGAP-GUIDED SELECTIVE DISTILLATION

Selective distillation extracts a plan from verified reasoning and allocates its supervision by relative student likelihood (Hsieh et al., 2023).

Extracting concise plans. An offline compressor f_ϕ receives a problem x and its verified teacher execution y^*, and constructs a plan $c^* = f_\phi(x, y^*)$. This compresses the completed trace into two or three high-level steps, excluding detailed calculations and the answer (Dou et al., 2026). Prepending c^* leaves the verified execution y^* unchanged. Although the compressor observes the completed execution, the student predicts the plan from x alone and then predicts y^* given x, c^*: $\mathcal{L}_{\text{distill}} = -\log \pi_\theta(c^* \mid x) - \log \pi_\theta(y^* \mid x, c^*)$. Matched Raw-SFT instead optimizes $-\log \pi_\theta(y^* \mid x)$.

Allocating plan supervision. Our selection variable is whether a verified execution receives a plan prefix, rather than whether the example is included in training (Chen et al., 2024; Xia et al., 2024). A frozen teacher scorer T and student scorer S score each extracted plan under the same problem. Let $\bar{\ell}_m(i)$ be its mean token log likelihood under m, and $\mathcal{E}_m(i) = -\bar{\ell}_m(i)$. Define

$$g_i = \bar{\ell}_T(i) - \bar{\ell}_S(i) = \mathcal{E}_S(i) - \mathcal{E}_T(i). \tag{1}$$

Lower-gap examples receive the plan prefix as a proxy for reachability and others retain the original verified target. Questions, executions, data order, and row count stay fixed. We compare quartiles with matched random allocation. Appendix E.1 provides the extraction and scoring details.

2.3 Plan-Conditioned Rollouts and Outcome Decomposition

With $G = KM$ terminal verifications, sample K plans $c_k \sim \pi_{\text{old}}(\cdot \mid x)$ and M independent executions $y_{k,m} \sim \pi_{\text{old}}(\cdot \mid x, c_k)$ per plan. Unlike inference-time search (Wang et al., 2023b; Yao et al., 2023a), these samples organize training feedback. For $R_{k,m} = R(x, y_{k,m})$, define the empirical plan value and the execution residual as

$$Q_k = \frac{1}{M}\sum_{m=1}^{M} R_{k,m}, \qquad D_{k,m} = R_{k,m} - Q_k. \tag{2}$$

Here Q_k estimates $q_{\text{old}}(x, c_k)$, while $D_{k,m}$ measures execution variation within a plan. With $\bar{Q} = K^{-1}\sum_k Q_k$, $R_{k,m} - \bar{Q} = (Q_k - \bar{Q}) + D_{k,m}$ separates differences across plans from differences among executions of the same plan. Standardizing the two terms separately gives

$$A^c_k = \frac{Q_k - \bar{Q}}{s_Q + \varepsilon_{\text{std}}}, \qquad A^y_{k,m} = \frac{D_{k,m}}{s_D + \varepsilon_{\text{std}}}. \tag{3}$$

s_Q is the sample deviation across plans, s_D across all KM residuals, and $\varepsilon_{\text{std}} > 0$ stabilizes normalization. Statistics are detached, $\sum_m D_{k,m} = 0$. Unlike a single group-relative signal (Shao et al., 2024; Ahmadian et al., 2024), the two advantages supervise distinct spans.

2.4 Verifier-Anchored Plan Contrast

The plan advantage summarizes execution outcomes at the plan level. Sibling comparisons refine its application across plan tokens. This adapts verifier-anchored contrast (Pan et al., 2026) to the plan span, alongside other forms of fine-grained reasoning feedback (Lightman et al., 2024; Cui et al., 2026; Zhao et al., 2026).

Constructing sibling comparisons. For target plan c_k, let $\mathcal{P}_k = \{j \neq k : c_j \text{ is format-valid}\}$ contain the valid non-target siblings. Format validity requires a nonempty, well-formed plan block. Select a highest-value sibling $j^+_k \in \arg\max_{j\in\mathcal{P}_k} Q_j$ and define

$$\mathcal{N}_k = \left\{ j \in \mathcal{P}_k \setminus \{j^+_k\} : Q_{j^+_k} - Q_j \geq \Delta \right\}, \qquad \Delta > 0. \tag{4}$$

Thus, negative contexts have lower empirical success. With frozen $\pi_{\text{snap}} = \pi_{\text{old}}$, score the target plan under a shared template $H(x, c_j)$ containing the problem and one sibling:

$$s_{k,t}(j) = \log \pi_{\text{snap}}(c_{k,t} \mid H(x, c_j), c_{k,<t}),$$

$$g_{k,t} = s_{k,t}(j^+_k) - \log\left(\frac{1}{|\mathcal{N}_k|}\sum_{j\in\mathcal{N}_k} e^{s_{k,t}(j)}\right). \tag{5}$$

Only the sibling changes across branches and the negative term averages probabilities before taking the logarithm. Hints contain no executions or reference solutions.

Modulating plan-token advantages. We transform the contrast into a bounded adjustment $d_{k,t} = \alpha \tanh(g_{k,t}/\tau_c)$, where $\alpha \geq 0$ controls its amplitude and $\tau_c > 0$ its scale. Let $I_{k,t} = 1$ when both comparison sets are available and $|d_{k,t}| > \eta$, with $\eta \geq 0$ and otherwise set $I_{k,t} = 0$. Unavailable contrasts have $d_{k,t} = 0$. The effective plan-token advantage is

$$\widetilde{A}^c_{k,t} = \text{sgn}(A^c_k)\max\{0,\ |A^c_k| + \text{sgn}(A^c_k) I_{k,t} d_{k,t}\}, \tag{6}$$

with $\text{sgn}(0) = 0$. It follows that $A^c_k \widetilde{A}^c_{k,t} \geq 0$ and $|\widetilde{A}^c_{k,t} - A^c_k| \leq \alpha$. Unmodulated plan tokens retain A^c_k, execution tokens retain $A^y_{k,m}$, and snapshot scores are detached.

2.5 PC-GRPO Training Objective

We use a GRPO surrogate (Shao et al., 2024) with PPO-style clipping. For token $z_{k,m,t}$ in $z_{k,m} = (c_k, y_{k,m})$, let $\rho_{k,m,t}(\theta) = \pi_\theta(z_{k,m,t} \mid x, z_{k,m,<t}) / \pi_{\text{old}}(z_{k,m,t} \mid x, z_{k,m,<t})$. For an advantage a, the token surrogate is

$$\ell_{k,m,t}(a) = \min\{\rho_{k,m,t} a,\ \text{clip}(\rho_{k,m,t}, 1-\epsilon, 1+\epsilon) a\}\,, \tag{7}$$

where $\epsilon > 0$ is the clipping radius. The objective is

$$\mathcal{L}_{\text{PC}} = -\mathbb{E}\left[\frac{1}{KM}\sum_{k,m}\left(\frac{\sum_{t\in\mathcal{T}_{c_k}} \ell_{k,m,t}(\widetilde{A}^{c}_{k,t})}{|\mathcal{T}_{c_k}|} + \frac{\sum_{t\in\mathcal{T}_{y_{k,m}}} \ell_{k,m,t}(A^{y}_{k,m})}{|\mathcal{T}_{y_{k,m}}|}\right)\right]. \tag{8}$$

The expectation covers problems and rollout groups and there is no reference-model KL penalty (Yu et al., 2025). A plan shared by M executions has a total plan-span weight of $1/K$ after averaging over those executions. Separate span averages prevent long executions from diluting plan updates (Yu et al., 2025; Liu et al., 2025). Replacing $\widetilde{A}^{c}_{k,t}$ with A^{c}_{k} yields the no-contrast ablation. Hints produce detached feedback and policy ratios use the original rollout context.

2.6 Computation and Inference

With conditionally independent executions, $\mathbb{E}[Q_k \mid x, c_k] = q_k$ and $\text{Var}(Q_k \mid x, c_k) = q_k(1-q_k)/M$, where $q_k = q_{\text{old}}(x, c_k)$. The K–M choice balances plan coverage and estimation within $G = KM$. Snapshot scoring uses one positive and $|\mathcal{N}_k|$ negative contexts per valid target plan, reused across its executions. Training requires no external teacher or learned critic and inference generates one plan and execution. Repeated execution and sibling scoring are confined to training. Appendix E provides additional estimator and implementation details.

3 Experiments

We evaluate selective plan supervision (RQ1), fixed-budget PC-GRPO (RQ2), plan-token contrast (RQ3), and accuracy versus training cost (RQ4). SFT and online studies use matched controls.

3.1 Experimental Setup

Models and data. For SFT, we train Qwen3-4B-Instruct-2507 from the Qwen3 family (Team, 2025) with two teacher-data systems: QwQ-32B/OpenThoughts3 (Guha et al., 2026) and GPT-OSS-120B/DASD (Yan et al., 2026). OpenThoughts3 trains separate 10K mathematics, code, and science checkpoints; DASD has two stages, a 64K training context, and 435K trajectories: approximately 105K low-temperature examples in Stage 1 and 330K high-temperature examples in Stage 2, with Stage 2 continuing from the Stage 1 checkpoint. Both stages are constructed with divergence-aware sampling, which selects full responses using teacher–student sequence-level discrepancies (Yan et al., 2026). Qwen2.5-32B-Instruct extracts plans. Matched Raw-SFT and Plan-SFT share questions, teacher traces, order, schedules, and context policies.

Main online comparisons use standard Qwen3 1.7B, 4B, 8B, and 14B checkpoints, plus Olmo-3-7B-Think as a non-Qwen reasoning architecture. Online runs use OpenThoughts Math 30K in one TRL/OPSD codebase with matched LoRA, sampling, and verification. All use $G = 8$ calls per problem; PC-GRPO uses $(K, M) = (4, 2)$, with a $(2, 4)$ ablation. The composition study tests three 4B initializations.

Comparisons and evaluation. We retain the local GRPO (Shao et al., 2024), OPSD (Zhao et al., 2026), RLCSD (Pan et al., 2026), and PTA-GRPO (Dou et al., 2026) comparisons. PTA-GRPO uses the common initialization rather than its published cold-start pipeline. We evaluate AMC23, AIME24/25, HMMT25, and held-out GPQA-Diamond (GPQA-D) and LiveCodeBench (LCB) after mathematics-only RL. Average@K denotes mean correctness over K sampled responses, not

pass@K. In the main SFT evaluation, AIME, LCB, and GPQA-D use Average@64, Average@6, and Average@3, respectively. In the RL evaluation, mathematics benchmarks use Average@12 and both OOD benchmarks use Average@3, all at temperature 1.0. AIME Avg. averages the AIME24 and AIME25 scores. DASD's standard generation cap is 38,912 tokens. All methods use identical evaluation settings. Appendix Table 9 additionally reports three independent 1.7B runs for Base, GRPO, RLCSD, and PC-GRPO.

3.2 PLAN-SFT: AUTOMATED PLAN SYNTHESIS

Table 1: Plan-SFT on Qwen3-4B-Instruct-2507. AIME, LCB, and GPQA-D use Average@64, Average@6, and Average@3. OpenThoughts3 uses domain-matched 10K checkpoints; DASD uses a 64K context. Long caps: 102,400 tokens (AIME), 81,920 (LCB/GPQA-D); standard cap: 38,912. Δ is Plan minus Raw; bold marks the better matched score. Mean tokens (AIME24/25, thousands) are a separate decoding diagnostic.

(a) QwQ-32B / OpenThoughts3: domain-matched 10K training subsets

Setting	AIME24	AIME25	AIME Avg.	Mean tok. (24/25)	LCB	LCB-Easy	LCB-Med.	LCB-Hard	GPQA-D
Stage1-Raw	59.30	57.50	58.40	13.94k/15.76k	44.85	**77.84**	34.63	13.14	48.15
Stage1-Plan	**66.20**	**63.50**	**64.80**	16.02k/16.88k	**58.32**	75.09	**60.60**	**29.67**	**56.90**
Δ	+6.90	+6.00	+6.40	+15.0%/ + 7.2%	+13.47	−2.75	+25.97	+16.53	+8.75

(b1) GPT-OSS-120B / DASD: long-budget evaluation

Setting	AIME24	AIME25	AIME Avg.	Mean tok. (24/25)	LCB	LCB-Easy	LCB-Med.	LCB-Hard	GPQA-D
Stage1-Raw	82.0	74.8	78.40	27.73k/32.66k	**58.41**	**85.44**	**55.28**	**23.68**	**63.42**
Stage1-Plan	**82.3**	**77.4**	**79.85**	27.43k/32.79k	52.84	81.04	47.69	19.72	61.08
Stage2-Raw	85.20	82.50	83.85	25.31k/29.47k	61.82	89.29	57.52	28.35	**64.30**
Stage2-Plan	**85.33**	**83.13**	**84.23**	26.14k/29.65k	**63.65**	**89.77**	**60.98**	**29.47**	63.32

(b2) GPT-OSS-120B / DASD: matched 38,912-token evaluation

Setting	AIME24	AIME25	AIME Avg.	Mean tok. (24/25)	LCB	LCB-Easy	LCB-Med.	LCB-Hard	GPQA-D
Stage1-Raw	74.30	66.53	70.42	20.14k/21.56k	**56.31**	**85.71**	**52.06**	**19.92**	**58.78**
Stage1-Plan	**74.60**	**67.13**	**70.87**	19.31k/21.45k	49.76	81.59	42.48	14.84	58.18
Stage2-Raw	**78.80**	73.17	75.98	18.61k/20.30k	60.98	**89.63**	56.67	25.81	**61.24**
Stage2-Plan	78.63	**73.70**	**76.17**	19.17k/21.41k	**62.08**	89.49	**58.37**	**27.74**	60.32

Table 2: Plan allocation on OpenThoughts3 Math10K (seed-42 avg@10). Sparse treatments augment 2,500 matched rows; Q1–Q4 ascend by PlanGap. **Ret.** is retained AIME gain versus full Plan-SFT; **ΔTok.** is pooled length change versus Raw-SFT.

Setting	Plan	AIME24	AIME25	Avg.	Mean tok. (24/25)	Gain/Raw	Ret.	ΔTok.
Base	0%	60.3	48.0	54.2	9.22k/7.21k	−4.9	–	−43.3%
Raw-SFT	0%	61.7	56.3	59.0	14.43k/14.53k	0.0	–	0.0%
Plan-SFT	100%	**66.7**	**64.7**	**65.7**	16.25k/16.96k	+6.7	100.0%	+14.7%
Random25-Plan-SFT	25%	64.0	59.7	61.9	15.38k/16.57k	+2.9	42.5%	+10.3%
PlanGap-Q1	25%	65.3	64.3	64.8	15.51k/16.84k	+5.8	86.6%	+11.7%
PlanGap-Q2	25%	49.0	51.7	50.4	11.89k/14.36k	−8.7	–	−9.4%
PlanGap-Q3	25%	49.7	53.0	51.4	13.02k/13.63k	−7.7	–	−8.0%
PlanGap-Q4	25%	59.3	60.7	60.0	13.94k/15.46k	+1.0	14.9%	+1.5%

Selective plans retain most of the full gain (RQ1). OpenThoughts3 Plan-SFT improves AIME Avg., LCB, and GPQA-D by 6.40, 13.47, and 8.75 points (Table 1). **PlanGap-Q1 retains 86.6% of full Plan-SFT's AIME gain with plans on 25% of rows**: 64.8 AIME Avg., just 0.9 below full augmentation and 3.0 above matched Random25 (Table 2). Its response length is 2.6% below full Plan-SFT but 11.7% above Raw-SFT. Q2–Q4 perform less well; low gap is useful for selection, not a monotonic difficulty rank. The detailed length and difficulty diagnostics are in Appendix D.

Temperature changes what plan supervision transfers. Low-temperature DASD Stage 1 concentrates on stable teacher modes, whereas Stage 2 adds diverse high-temperature trajectories (Yan et al., 2026). Plan-SFT improves AIME Avg. in both stages; Stage 1 reduces LCB and GPQA-D,

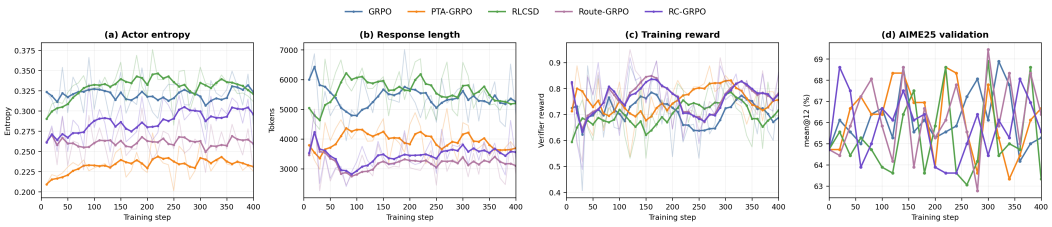

Figure 3: Qwen3-4B training dynamics over 400 matched optimizer steps. Panels (a–c) show raw logs and trailing five-point means; panel (d) shows saved-checkpoint AIME25 Average@12 without smoothing.

while Stage 2 improves LCB but still slightly reduces GPQA-D (Table 1). Across decoding seeds, Stage2-Plan also has narrower AIME ranges than Stage2-Raw at both budgets (Appendix Tables 13 and 14), consistent with compact plans stabilizing diverse executions. DASD selects full responses, whereas PlanGap selects student-reachable plan spans; Q1 therefore retains most of the Plan-SFT gain with only 25% plan augmentation.

3.3 PC-GRPO: Main Results

Table 3: Post-training in the matched OPSD/TRL framework. Mathematics uses Average@12; GPQA-D and LCB use Average@3 and assess transfer after mathematics-only RL. AIME Avg. averages AIME24/25. Δ denotes the absolute change in AIME Avg. from the corresponding Base model, in percentage points. Bold/underline: best/second per model. Time/GRPO: 400-step wall time relative to same-size GRPO.

Model	Method	Mathematics					OOD transfer		Δ	Time/GRPO ↓
		AMC23	AIME24	AIME25	HMMT25	AIME Avg.	GPQA-D	LCB		
Qwen3-1.7B	Base	85.21	46.39	36.67	21.11	41.53	48.99	55.25	–	–
	GRPO	86.04	51.11	37.50	25.28	44.31	49.33	53.42	+2.78	1.000×
	OPSD	86.46	51.94	39.44	**27.78**	45.69	49.83	55.71	+4.16	–
	RLCSD	86.04	53.33	38.89	27.22	46.11	49.16	55.12	+4.58	1.132×
	PTA-GRPO (K4M2)	86.67	52.50	38.89	26.11	45.70	**51.18**	**56.43**	+4.17	0.821×
	PC-GRPO (K4M2)	**87.92**	**54.44**	**40.83**	25.56	**47.64**	50.84	**56.43**	**+6.11**	**0.735×**
Qwen3-4B	Base	96.88	73.89	65.56	42.78	69.73	57.91	**77.76**	–	–
	GRPO	97.92	75.56	67.78	45.83	71.67	56.57	75.99	+1.94	1.000×
	OPSD	97.50	75.83	65.83	45.56	70.83	59.09	75.54	+1.10	–
	RLCSD	**98.75**	76.11	**68.89**	45.28	72.50	59.43	76.06	+2.77	1.083×
	PTA-GRPO (K4M2)	98.12	76.11	67.22	46.67	71.67	58.92	77.23	+1.94	0.760×
	PC-GRPO (K4M2)	98.33	**76.94**	68.61	**47.22**	**72.78**	**59.60**	76.91	**+3.05**	**0.702×**
Qwen3-8B	Base	95.62	73.89	67.22	44.72	70.56	65.99	64.64	–	–
	GRPO	96.46	77.78	71.39	48.06	74.59	64.81	64.71	+4.03	1.000×
	OPSD	95.42	73.33	66.11	45.00	69.72	64.98	58.84	−0.84	–
	RLCSD	**97.50**	78.06	70.83	**48.89**	74.45	65.49	64.97	+3.89	1.106×
	PTA-GRPO (K4M2)	97.08	78.06	70.56	46.39	74.31	65.15	**66.14**	+3.75	0.805×
	PC-GRPO (K4M2)	97.29	**78.89**	**71.94**	48.06	**75.42**	**66.84**	65.69	**+4.86**	**0.693×**
Qwen3-14B	Base	99.17	79.44	73.89	50.28	76.67	65.49	78.73	–	–
	GRPO	99.17	80.56	74.17	50.83	77.37	**67.17**	78.67	+0.70	1.000×
	RLCSD	**99.38**	81.11	**75.00**	**53.06**	**78.06**	65.99	79.52	**+1.39**	1.043×
	PTA-GRPO (K4M2)	98.96	80.56	74.17	48.89	77.37	65.82	78.15	+0.70	0.791×
	PC-GRPO (K4M2)	99.17	**81.39**	74.72	52.78	**78.06**	65.82	**79.78**	**+1.39**	**0.697×**
Olmo-3-7B-Think	Base	95.00	72.78	63.61	41.11	68.20	53.03	68.10	–	–
	GRPO	96.88	74.44	67.78	**46.94**	71.11	52.69	**69.08**	+2.91	1.000×
	RLCSD	97.29	73.33	68.33	46.39	70.83	52.86	67.25	+2.63	1.052×
	PTA-GRPO (K4M2)	97.29	75.00	67.50	46.11	71.25	54.38	68.62	+3.05	0.952×
	PC-GRPO (K4M2)	**97.50**	**75.28**	**69.72**	46.67	**72.50**	**56.73**	68.69	**+4.30**	**0.854×**

PC-GRPO improves AIME Avg. across families (RQ2). PC-GRPO exceeds matched GRPO by 3.33, 1.11, 0.83, and 0.69 points across Qwen3 1.7B–14B, respectively (Table 3). It leads at 1.7B–8B and ties RLCSD at 14B. Olmo-3-7B-Think reaches 72.50, 1.39 points above GRPO.

Multi-seed robustness. Across three 1.7B runs, PC-GRPO has the highest AIME24/25 means (51.39/38.24), ahead of GRPO (51.02/37.13) and RLCSD (49.72/35.18). Its larger AIME25 deviation (4.32) nevertheless shows remaining run-to-run variance (Appendix Table 9).

Table 4: **Core PC-GRPO ablations.** (a) Qwen3-1.7B varies contrast and the allocation of eight verifier calls. (b) Qwen3-4B varies contrast scale at K4M2.

(a) Contrast and rollout allocation (1.7B)

Condition	AIME24	AIME25	Avg.
PC-GRPO w/o contrast (K2M4)	52.50	39.44	45.97
PC-GRPO w/o contrast (K4M2)	54.17	38.89	46.53
PC-GRPO (K4M2)	**54.44**	**40.83**	**47.64**

(b) Contrast scale (4B, K4M2)

Scale	AIME24	AIME25	Avg.
0.25	75.56	**68.61**	72.09
0.50	**76.94**	**68.61**	**72.78**
0.75	76.11	68.33	72.22

Table 5: Cross-scale plan-contrast ablation. Removing contrast preserves $K \times M$ sampling and separate plan/execution advantages.

Model	Condition	AIME24	AIME25	AIME Avg.
Qwen3-1.7B	PC-GRPO w/o contrastive reward (K4M2)	54.17	38.89	46.53
	PC-GRPO w/o contrastive reward (K2M4)	52.50	39.44	45.97
	PC-GRPO (K4M2)	**54.44**	**40.83**	**47.64**
Qwen3-4B	PC-GRPO w/o contrastive reward (K4M2)	76.39	**69.44**	**72.92**
	PC-GRPO w/o contrastive reward (K2M4)	**76.94**	68.89	**72.92**
	PC-GRPO (K4M2)	**76.94**	68.61	72.78
Qwen3-8B	PC-GRPO w/o contrastive reward (K4M2)	78.06	70.83	74.45
	PC-GRPO w/o contrastive reward (K2M4)	78.61	**73.06**	**75.84**
	PC-GRPO (K4M2)	**78.89**	71.94	75.42
Qwen3-14B	PC-GRPO w/o contrastive reward (K4M2)	80.56	73.61	77.09
	PC-GRPO (K4M2)	**81.39**	**74.72**	**78.06**
Olmo-3-7B	PC-GRPO w/o contrastive reward (K4M2)	**76.67**	68.61	**72.64**
	PC-GRPO (K4M2)	75.28	**69.72**	72.50

Transfer is strongest in science, mixed in code. PC-GRPO leads GPQA-D at 4B, 8B, and Olmo-3 after mathematics-only training. LCB gains vary: PC-GRPO ties for best at 1.7B, leads at 14B, but the 4B Base remains strongest. The per-model scores, including OPSD, are in Table 3.

3.4 PC-GRPO: Value of Plan-Token Contrast

At 1.7B, plan-token contrast adds 1.11 AIME points over matched K4M2 without contrast (Table 4). K2M4 without contrast scores 0.56 points below K4M2, isolating plan allocation. The 4B scale sweep peaks at 0.50. Across models, the contrast effect varies (Table 5).

Contrast effects vary across scales. At matched K4M2, contrast changes AIME Avg. by $+1.11$, -0.14, $+0.97$, $+0.97$, and -0.14 points across the five models (Table 5). The gains in three settings show a useful refinement, while the two small reversals bound its empirical claim. K2M4 also changes outcomes independently of contrast.

Plan scalability. With four executions per plan, $K = 6$ gives the best AIME Avg. (43.33), compared with 42.69 at $K = 4$ and 41.67 at $K = 8$. More sampled plans therefore do not monotonically improve accuracy (Appendix Table 10).

Token placement matters at the fixed 4B checkpoint. Permuting plan weights while preserving their values and signs reduces AIME Avg. from 70.28 to 67.92; aligned PC-GRPO restores 2.36 points but matches flat credit (Table 6a).

Plan supervision composes with PC-GRPO. All three initializations improve after PC-GRPO (Table 6b), by 3.20, 1.67, and 2.78 AIME points for Instruct, DASD Raw, and DASD Plan, respectively. The Plan branch finishes 1.94 points above the Raw branch. This single-run comparison establishes compatibility, not statistical synergy.

3.5 Computational Efficiency and Scope

PC-GRPO uses 0.693**–**$0.854\times$ **GRPO wall time** across Qwen3 and Olmo-3, versus 1.043–$1.132\times$ for RLCSD (Table 3). Figure 3 shows matched 4B training dynamics; Appendix Figure 5 gives the accuracy–cost comparison. These hardware-dependent times are not FLOP estimates.

Table 6: **PC-GRPO controls and composition on 4B models.** (a) K4M2 credit controls with separate span averages; each Δ is relative to the preceding row. (b) PC-GRPO after three initializations; bold marks pairwise gains. Seed-42 Average@12.

(a) Plan-local credit controls

Condition	AIME24	AIME25	Avg.	Δ prev.
Flat credit	75.00	65.56	70.28	–
PC-GRPO w/o contrastive reward	72.50	65.28	68.89	-1.39
PC, permuted plan weights	71.67	64.17	67.92	-0.97
PC-GRPO	75.56	65.00	70.28	$+2.36$

(b) SFT initialization → PC-GRPO

Initialization	AMC23	AIME24	AIME25	HMMT25	AIME Avg.
Instruct-2507	94.38	62.50	47.22	30.00	54.86
+ PC-GRPO	**95.42**	**64.72**	**51.39**	**33.33**	**58.06**
DASD Stage2-Raw	98.12	77.50	72.78	62.78	75.14
+ PC-GRPO	**98.75**	**79.44**	**74.17**	**63.33**	**76.81**
DASD Stage2-Plan	97.71	80.00	71.94	62.78	75.97
+ PC-GRPO	**98.33**	**81.39**	**76.11**	**64.17**	**78.75**

4 Related Work

Plan distillation and structured feedback. Sequence distillation transfers teacher reasoning (DeepSeek-AI, 2025; Muennighoff et al., 2025; Ye et al., 2025). OpenThoughts scales data construction (Guha et al., 2026), while DASD selects full sequences (Yan et al., 2026). Planning templates expose structure (Parmar et al., 2025; Yang et al., 2024), and PTA-GRPO adds plan quality to a response-level reward (Dou et al., 2026). Other group-based objectives derive response-level updates from sampled outputs (Zhang et al., 2026a; 2025). Process supervision, prefix values, and hierarchical methods provide finer feedback (Lightman et al., 2024; Wang et al., 2024; Kazemnejad et al., 2025; Li et al., 2026; Tran et al., 2026; Xiong et al., 2026; Wang et al., 2026; Feng et al., 2025; Peng et al., 2026). PlanGap selects supervision by the teacher–student mean plan NLL gap, while PC-GRPO assigns plan and execution feedback to separate spans.

On-policy distillation. On-policy distillation scores student-generated trajectories (Agarwal et al., 2024; Zhao et al., 2026; Tan & Hong, 2026; Zhang et al., 2026b). Providing gold solutions to the teacher can suppress self-correction in thinking models (Kaur et al., 2026). RLCSD contrasts correct and incorrect solution hints (Pan et al., 2026), while CSCR uses counterfactual sensitivity to redistribute feedback (He et al., 2026). PC-GRPO ranks sibling plans using repeated verifier outcomes and confines token-level contrast to the plan span.

5 Conclusion

We investigate compact plans as distinct learning targets in supervised distillation and reinforcement learning. PlanGap-guided distillation prioritizes plans that are more accessible to the student while preserving verified executions. Independently, PC-GRPO uses repeated executions under a fixed verifier budget to distinguish plan reliability from execution-specific outcomes, without an SFT cold start. Comparisons among plans with different empirical success rates modulate plan-token advantages while preserving the direction of verifier-derived feedback. Selective augmentation retains most of the benefit of full plan supervision with fewer plan-annotated examples, while PC-GRPO improves on matched GRPO with less measured training time. Compact plans thereby provide an explicit learning target for selective distillation and plan-aware policy optimization.

AI USE STATEMENT

Generative AI tools were used to refine the Plan-SFT compression prompt, assist with baseline code adaptation and reproduction, draft scientific figures, and support literature review, result interpretation, and manuscript editing. All AI-assisted code, data, figures, claims, and reported results were reviewed and verified by the authors. We take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Section 2 defines the objectives and estimator conventions. Appendices A–E provide the plan extraction instruction, an illustrative trace, implementation aliases, supplementary completed results, and expanded optimization details. Our code is publicly available at `https://anonymous.4open.science/r/PC-GRPO`.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=3zKtaqxLhW.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce-style optimization for learning from human feedback in llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 12248–12267. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.662. URL https://doi.org/10.18653/v1/2024.acl-long.662.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. Alpagasus: Training a better alpaca with fewer data. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=FdVXgSJhvz.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, Jiarui Yuan, Huayu Chen, Kaiyan Zhang, Xingtai Lv, Shuo Wang, Yuan Yao, Xu Han, Hao Peng, Yu Cheng, Zhiyuan Liu, Maosong Sun, Bowen Zhou, and Ning Ding. Process reinforcement through implicit rewards. *Trans. Mach. Learn. Res.*, 2026, 2026. URL https://openreview.net/forum?id=9SkkifLopZ.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948, 2025. doi: 10.48550/ARXIV.2501.12948. URL https://doi.org/10.48550/arXiv.2501.12948.

Zhihao Dou, Qinjian Zhao, Zhongwei Wan, Zhang Dinggen, Weida Wang, Benteng Chen, Towsif Raiyan, Qingtao Pan, Yang Ouyang, Chaoda Song, Zhiqiang Gao, Shufei Zhang, and Sumon Biswas. Plan then action: High-level planning guidance reinforcement learning for LLM reasoning. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=TTuzGDlGNA.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/420c9f777c0b4f78d515e53cf74d58b2-Abstract-Conference.html.

Etash Kumar Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Rea Sprague, Ashima Suvarna, Benjamin Feuer, Leon Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, Wanjia Zhao, John Yang, Shreyas Pimpalgaonkar, Kartik sharma, Charlie

Cheng-Jie Ji, Yichuan Deng, Sarah M Pratt, Vivek Ramanujan, Jon Saad-Falcon, Stutee Acharya, Jeffrey Li, Achal Dave, Alon Albalak, Kushal Arora, Blake Wulfe, Chinmay Hegde, Greg Durrett, Sewoong Oh, Mohit Bansal, Saadia Gabriel, Aditya Grover, Kai-Wei Chang, Vaishaal Shankar, Aaron Gokaslan, Mike A Merrill, Tatsunori Hashimoto, Yejin Choi, Jenia Jitsev, Reinhard Heckel, Maheswaran Sathiamoorthy, Alex Dimakis, and Ludwig Schmidt. Openthoughts: Data recipes for reasoning models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=7xjoTuaNmN.

Qiangqiang He, Zhongheng Wu, and Zijian Wang. Not all tokens deserve equal credit: Counterfactual sensitivity credit reallocation for long-CoT reasoning. *arXiv preprint arXiv:2607.27888*, 2026. URL https://arxiv.org/abs/2607.27888.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, volume ACL 2023 of *Findings of ACL*, pp. 8003–8017. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-ACL.507. URL https://doi.org/10.18653/v1/2023.findings-acl.507.

Simran Kaur, Narutatsu Ri, Yinghui He, Liam H Fowl, and Sanjeev Arora. Rethinking on-policy self-distillation for thinking models. In *COLM 2026 Workshop on Efficient Reasoning*, 2026. URL https://openreview.net/forum?id=NmHpi2SCtC.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron C. Courville, and Nicolas Le Roux. Vineppo: Refining credit assignment in RL training of llms. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/kazemnejad25a.html.

Yuetai Li, Xiang Yue, Zhangchen Xu, Fengqing Jiang, Luyao Niu, Bill Yuchen Lin, Bhaskar Ramasubramanian, and Radha Poovendran. Small models struggle to learn from strong reasoners. In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, volume ACL 2025 of *Findings of ACL*, pp. 25366–25394. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.FINDINGS-ACL.1301. URL https://doi.org/10.18653/v1/2025.findings-acl.1301.

Yugu Li, Zehong Cao, Jianglin Qiao, and Siyi Hu. SSVPO: Effective step-level credit assignment for RL training of language models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=g33DGvnHYd.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

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.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel J. Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 20275–20321. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.EMNLP-MAIN.1025. URL https://doi.org/10.18653/v1/2025.emnlp-main.1025.

Leyi Pan, Shuchang Tao, Yunpeng Zhai, Lingzhe Zhang, Zhaoyang Liu, Bolin Ding, Aiwei Liu, and Lijie Wen. RLCSD: Reinforcement learning with contrastive on-policy self-distillation. *arXiv preprint arXiv:2606.11709*, 2026. URL https://arxiv.org/abs/2606.11709.

Mihir Parmar, Palash Goyal, Xin Liu, Yiwen Song, Mingyang Ling, Chitta Baral, Hamid Palangi, and Tomas Pfister. PLAN-TUNING: post-training language models to learn step-by-step planning for complex problem solving. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 21419–21433. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.EMNLP-MAIN.1087. URL https://doi.org/10.18653/v1/2025.emnlp-main.1087.

Jiangweizhi Peng, Yuanxin Liu, Ruida Zhou, Charles Fleming, Zhaoran Wang, Alfredo Garcia, and Mingyi Hong. HiPER: Hierarchical plan–execute RL for multi-turn LLM agents. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=RJWQcgd3hX.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/ARXIV.2402.03300. URL https://doi.org/10.48550/arXiv.2402.03300.

Zhiquan Tan and Yinrong Hong. Self-supervised on-policy distillation for reasoning language models. *CoRR*, abs/2605.17497, 2026. doi: 10.48550/ARXIV.2605.17497. URL https://doi.org/10.48550/arXiv.2605.17497.

Qwen Team. Qwen3 technical report. *CoRR*, abs/2505.09388, 2025. doi: 10.48550/ARXIV.2505.09388. URL https://doi.org/10.48550/arXiv.2505.09388.

Hieu Tran, Zonghai Yao, and Hong Yu. Exploiting tree structure for credit assignment in reinforcement learning with large language models. In *Findings of the Association for Computational Linguistics, ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 10795–10810. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.FINDINGS-ACL.524. URL https://doi.org/10.18653/v1/2026.findings-acl.524.

Haozhe Wang, Qixin Xu, Che Liu, Junhong Wu, Fangzhen Lin, and Wenhu Chen. Emergent hierarchical reasoning in LLMs through reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=NlkykTqAId.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pp. 2609–2634. Association for Computational Linguistics, 2023a. doi: 10.18653/V1/2023.ACL-LONG.147. URL https://doi.org/10.18653/v1/2023.acl-long.147.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 9426–9439. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.510. URL https://doi.org/10.18653/v1/2024.acl-long.510.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023b. URL https://openreview.net/forum?id=1PL1NIMMrw.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

Wei Wu, Liyi Chen, Congxi Xiao, Tianfu Wang, Qimeng Wang, Chengqiang Lu, Yan Gao, Yi Wu, Yao Hu, and Hui Xiong. Anti-length shift: Dynamic outlier truncation for training efficient reasoning models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 22863–22891. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.ACL-LONG.1047. URL https://doi.org/10.18653/v1/2026.acl-long.1047.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: selecting influential data for targeted instruction tuning. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54104–54132. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/xia24c.html.

Siheng Xiong, Ali Payani, and Faramarz Fekri. Enhancing language model reasoning with structured multi-level modeling. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=PlkzZhqBCd.

Shaotian Yan, Kaiyuan Liu, Chen Shen, Bing Wang, Sinan Fan, Jun Zhang, Yue Wu, Zheng Wang, and Jieping Ye. Distribution-aligned sequence distillation for superior long-cot reasoning. *CoRR*, abs/2601.09088, 2026. doi: 10.48550/ARXIV.2601.09088. URL https://doi.org/10.48550/arXiv.2601.09088.

Ling Yang, Zhaochen Yu, Tianjun Zhang, Minkai Xu, Joseph E. Gonzalez, Bin Cui, and Shuicheng Yan. Supercorrect: Supervising and correcting language models with error-driven insights. *CoRR*, abs/2410.09008, 2024. doi: 10.48550/ARXIV.2410.09008. URL https://doi.org/10.48550/arXiv.2410.09008.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023a. URL http://papers.nips.cc/paper_files/paper/2023/hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract-Conference.html.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023b. URL https://openreview.net/forum?id=WE_vluYUL-X.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. LIMO: less is more for reasoning. *CoRR*, abs/2502.03387, 2025. doi: 10.48550/ARXIV.2502.03387. URL https://doi.org/10.48550/arXiv.2502.03387.

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 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/a4277440d50f1f15d2cb4c14f7e0c0d2-Abstract-Conference.html.

Kaichen Zhang, Shenghao Gao, Yuzhong Hong, Haipeng Sun, Junwei Bao, Hongfei Jiang, Yang Song, Hong Dingqian, and Hui Xiong. Rspo: Risk-seeking policy optimization for pass@ k and max@ k metrics in large language models. *arXiv preprint arXiv:2508.01174*, 2025.

Kaichen Zhang, Yuzhong Hong, Junwei Bao, Hongfei Jiang, Yang Song, Hong Dingqian, and Hui Xiong. Gvpo: Group variance policy optimization for large language model post-training. *Advances in Neural Information Processing Systems*, 38:165798–165820, 2026a.

Kaichen Zhang, Yuzhong Hong, Junwei Bao, Hongfei Jiang, Yang Song, Dingqian Hong, and Hui Xiong. Gvpo++: Group variance policy optimization for llm post-training and on-policy distillation. *arXiv preprint arXiv:2609.21432*, 2026b.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=Jpxfof0EaS.

A PLAN EXTRACTION PROMPT AND VALIDATION

The compressor receives both the problem and a verified reasoning path. The central instruction is:

> Compress the detailed reasoning path into exactly two or three numbered, high-level strategic steps. Summarize groups of reasoning actions rather than individual calculations. Ignore preliminary mistakes and retain only the final successful line of reasoning. Preserve logical order; allow essential variables, equations, and constraints; do not include detailed arithmetic, verification narration, or the final answer. Do not add <plan> tags.

The builder strips accidental wrappers, verifies the step count and non-empty content, and preserves row identity. Malformed generations fall back to a matched Raw-SFT row rather than silently dropping the example. This prompt and validation policy define the Plan-SFT construction used in our experiments.

B ILLUSTRATIVE PLAN-SFT REASONING TRACE

Figure 4 revisits the motivating geometry example using the terminology and mechanism of the current paper. It isolates the role of the plan prefix: the right-hand trace first commits to an exact geometric strategy and then executes it. The text is abbreviated and typeset for legibility, so the figure should be read as a schematic rendering of the reasoning process rather than a verbatim benchmark transcript.

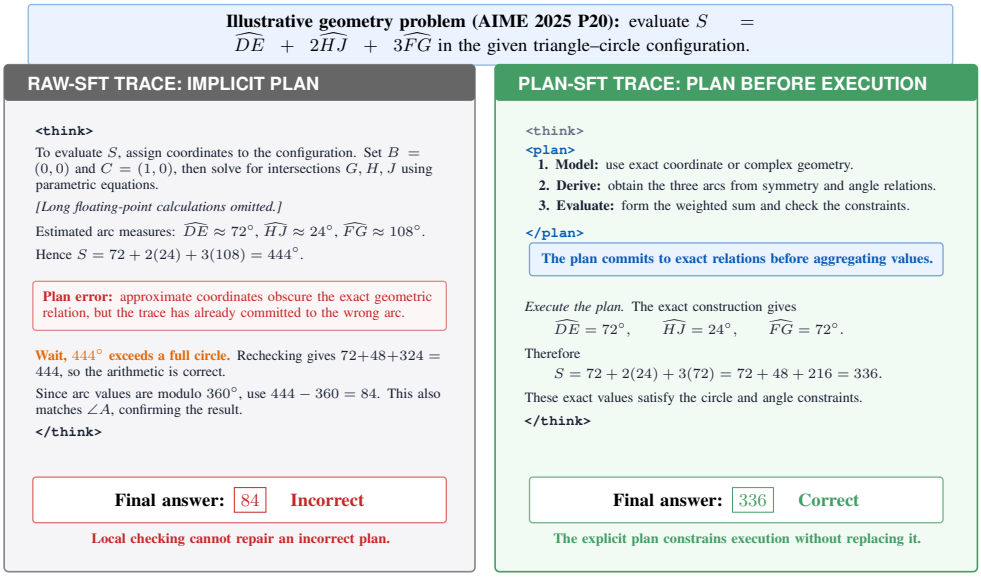

Figure 4: **Illustrative Plan-SFT geometry trace.** In the flat trace (left), an early approximate-coordinate choice produces an incorrect arc and subsequent checking only validates the arithmetic. In the plan-structured trace (right), a concise <plan> commits to exact geometric relations before detailed execution. Text is abbreviated for readability.

C METHOD AND IMPLEMENTATION MAPPING

Table 7 summarizes the learning signal assigned to each token span. Plan-SFT and PC-GRPO share the same <plan> interface but are evaluated independently; the main PC-GRPO experiments do not require a supervised cold start.

Table 7: Plan and execution signals used by each training condition.

Method	Regime	Plan signal	Execution signal
Raw-SFT	Offline	Implicit	Teacher tokens
Plan-SFT	Offline	Compressed target	Teacher tokens
GRPO	Online	Shared outcome	Shared outcome
PC-GRPO w/o contrastive reward	Online	Plan value	Within-plan residual
PC-GRPO	Online	Contrastive plan value	Within-plan residual

Historical scripts and checkpoints use the aliases in Table 8; these names are reported only to map the released artifacts to the paper terminology.

Table 8: Paper names and historical implementation aliases.

Paper name	Implementation alias
PC-GRPO w/o contrastive reward	RCPT-v1
PC-GRPO	RCPT-v2

D Existing Supplementary Results

This appendix separates planned robustness and scalability protocols from completed supplementary measurements. Blank cells mark pending measurements rather than zero-valued results. The DASD seed tables below report decoding repetitions of fixed checkpoints rather than independent training runs. External DASD references provide context separately from matched Raw/Plan comparisons.

The complete matched SFT table is retained in the main text (Table 1).

Additional SFT diagnostics. OpenThoughts3 LCB gains concentrate on medium and hard problems; its easy split and CodeForces decrease. DASD Stage 1 improves AIME while reducing code and science scores at both evaluation caps. DASD Stage 2 gains 0.19 AIME-average and 1.10 LCB points at 38,912 tokens, and 0.38/1.83 at long budgets, while GPQA-D falls by 0.92/0.98. In a separate 10-rollout-per-problem length diagnostic, OpenThoughts3 outputs lengthen by 15.0%/7.2% on AIME24/25 (10.8% pooled) under the 32,768-token cap. DASD Stage 1 pooled lengths change by $-2.3\%/-0.3\%$ at standard/long caps; Stage 2 changes by $+4.3\%/+1.8\%$. These length measurements do not recompute accuracy. PlanGap-Q1's pooled mean length is 16.18K, 2.6% below full Plan-SFT and 11.7% above Raw-SFT; Q2/Q3 lose 8.7/7.7 accuracy points despite shorter responses, and Q4 retains 14.9% of the full gain. Per augmented row, Q1 attains $3.46\times$ full augmentation's AIME gain efficiency in this matched diagnostic.

D.1 Planned Robustness and Scalability Evaluations

We reserve the following matched protocols for robustness and scaling measurements. They are intentionally left blank until every run in a comparison is complete.

Table 9: Planned Qwen3-1.7B multi-seed evaluation on AIME24 and AIME25. For trained methods, seeds 42–44 denote independent training runs followed by matched evaluation; for Base, they denote matched independent evaluations of the same checkpoint. Mean and standard deviation will be reported only after all three seeds are complete.

Method	Dataset	Seed 42	Seed 43	Seed 44	Mean ± Std.
Base	AIME24	45.08	47.37	47.23	46.56 ± 1.28
	AIME25	36.35	38.05	38.20	37.53 ± 1.03
GRPO	AIME24	51.11	51.11	50.83	51.02 ± 0.16
	AIME25	36.67	37.78	36.94	37.13 ± 0.58
RLCSD	AIME24	47.22	51.39	50.56	49.72 ± 2.21
	AIME25	33.61	34.44	37.50	35.18 ± 2.05
PC-GRPO (K4M2)	AIME24	51.67	53.06	49.44	51.39 ± 1.83
	AIME25	34.17	42.78	37.78	38.24 ± 4.32

Table 10: Planned Qwen3-1.7B PC-GRPO scalability evaluation. The number of executions per plan is fixed at $M = 4$, while the number of sampled plans grows from $K = 4$ to $K = 8$. Thus, this study scales strategic diversity and records the corresponding increase in total verifier calls $G = K \times M$.

K	M	G	AIME24	AIME25	Avg.
4	4	16	48.89	36.48	42.69
6	4	24	49.26	37.41	43.33
8	4	32	47.96	35.37	41.67

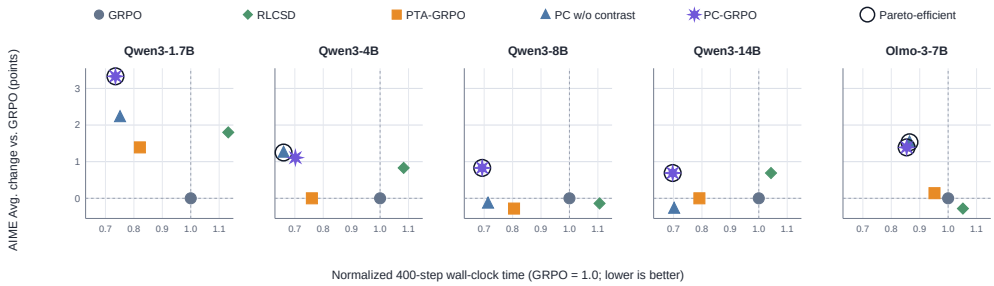

Figure 5: Accuracy–cost view of the completed post-training runs. Each point uses the AIME24/25 average from Table 3 and 400-step wall-clock time normalized by same-size GRPO. The dashed line denotes the GRPO cost baseline; the shaded region is the lower-cost, higher-accuracy quadrant relative to GRPO. Lines connect methods only to aid reading and do not imply a continuous frontier.

Table 11: Secondary cross-domain Plan-SFT evaluations. These completed results are retained alongside LiveCodeBench; they show mixed code outcomes.

Benchmark	Raw-SFT	Plan-SFT	Δ
CodeElo	24.64 ± 0.60	25.06 ± 1.33	+0.42
CodeForces	25.09 ± 0.96	21.78 ± 5.39	−3.31
JEEBench	69.95 ± 2.06	70.44 ± 1.22	+0.49

Table 12: External DASD reference points at the 102,400-token AIME inference budget. The Stage 1 row is reported by the DASD paper; DASD-4B-Thinking is our evaluation of the released checkpoint, averaged over decoding seeds 42–44. These rows provide context but are excluded from the matched Raw/Plan comparisons in Table 1.

Reference	AIME24	AIME25	Avg.
DASD Stage 1 report	84.20	74.00	79.10
DASD-4B-Thinking	84.07	82.60	83.33

Table 13: Seed-level DASD Stage 2 results at a 102,400-token inference budget. These are decoding seeds for one checkpoint per setting.

Setting / seed	AIME24	AIME25	Avg.
DASD-4B-Thinking / 42	85.4	82.4	83.90
DASD-4B-Thinking / 43	83.9	83.2	83.55
DASD-4B-Thinking / 44	82.9	82.2	82.55
Stage2-Raw / 42	84.3	82.8	83.55
Stage2-Raw / 43	86.1	82.6	84.35
Stage2-Raw / 44	85.2	82.1	83.65
Stage2-Plan / 42	85.2	82.9	84.05
Stage2-Plan / 43	84.8	83.4	84.10
Stage2-Plan / 44	86.0	83.1	84.55

Table 14: Seed-level DASD results at a 38,912-token inference budget. All rows use Average@64 with temperature 1.0 and top-p 1.0. These are decoding seeds from one checkpoint per setting, not independent training seeds. The reported harness scores are retained unchanged; two of the 24 benchmark–seed runs contain one empty sample.

Stage	Setting / seed	AIME24	AIME25	Avg.
1	Raw / 42	74.1	65.5	69.80
	Raw / 43	74.1	67.8	70.95
	Raw / 44	74.7	66.3	70.50
	Plan / 42	74.0	68.5	71.25
	Plan / 43	74.7	66.8	70.75
	Plan / 44	75.1	66.1	70.60
2	Raw / 42	78.3	73.0	75.65
	Raw / 43	78.2	73.6	75.90
	Raw / 44	79.9	72.9	76.40
	Plan / 42	78.3	73.9	76.10
	Plan / 43	78.3	74.3	76.30
	Plan / 44	79.3	72.9	76.10

E Detailed Estimators and Optimization

This appendix expands Section 2 using the same notation: c is a plan, y an execution, and R the terminal verifier reward.

E.1 Plan Representation and SFT Construction

The response factorization is $\pi_\theta(c, y \mid x) = \pi_\theta(c \mid x)\pi_\theta(y \mid x, c)$. The `<plan>` boundary defines disjoint token sets $\mathcal{T}_c$ and $\mathcal{T}_y$ but does not forbid later revision. For a verified teacher execution y^*, the compressor constructs

$$c^* = f_\phi(x, y^*). \tag{9}$$

The supervised target is

$$\texttt{<think><plan>}\ c^*\ \texttt{</plan>}\ y^*\ \texttt{</think>}. \tag{10}$$

The compressor observes the completed trace, but the student predicts c^* from x alone. The original y^* is preserved, including exploration or correction already present. Format checks enforce nonempty content and the required step count; they do not by themselves certify semantic faithfulness. The complete instruction appears in Appendix A.

For sparse allocation, the already aligned 10K Raw-SFT and Plan-SFT files are reused: no plan is regenerated and no row is dropped. Under one canonical Qwen ChatML template, both QwQ-32B (T) and Qwen3-4B-Instruct-2507 (S) receive the problem followed by `<think><plan>`; only the inner plan c_i^* is teacher-forced. Each model uses its own tokenization, and its score is divided by its corresponding number of plan tokens:

$$\bar{\ell}_m(i) = \frac{1}{|c_i^*|_m} \sum_t \log p_m(c_{i,t}^* \mid x_i, c_{i,<t}^*). \tag{11}$$

The raw directional PlanGap used for the reported partition is

$$g_i = \bar{\ell}_T(i) - \bar{\ell}_S(i) = \mathcal{E}_S(i) - \mathcal{E}_T(i). \tag{12}$$

It is not an absolute difference. The 10K examples are stably sorted in ascending g_i and split globally into four equal quartiles. PlanGap-Q1 thus contains the smallest student–teacher mean-NLL gaps and PlanGap-Q4 the largest. This ordering measures relative reachability, not plan correctness or intrinsic difficulty: a low gap says that the student assigns probability closer to the teacher, while the verified source trace supplies the external quality control. The compressed plan was produced by Qwen2.5-32B-Instruct rather than QwQ-32B: the QwQ score measures compatibility with the extracted plan, while the student score measures its reachability. Each sparse treatment selects exactly 2,500 paired Plan-SFT rows and retains 7,500 Raw-SFT rows. Questions, verified executions, input order, and row count are identical across treatments; Random25 uses the same ratio under seeded uniform sampling.

E.2 Repeated-Execution Estimator

For each problem,

$$c_k \sim \pi_{\text{old}}(\cdot \mid x), \qquad y_{k,m} \sim \pi_{\text{old}}(\cdot \mid x, c_k), \qquad R_{k,m} = R(x, c_k, y_{k,m}). \tag{13}$$

Every execution in group k has exactly the same sampled plan prefix. The decomposition

$$R_{k,m} - \bar{Q} = (Q_k - \bar{Q}) + (R_{k,m} - Q_k), \qquad \sum_m (R_{k,m} - Q_k) = 0 \tag{14}$$

motivates separately normalized advantages A_k^c and $A_{k,m}^y$ in Eq. (3). Rewards and normalization statistics are detached during policy updates. If all plan values are identical, plan advantages are zero; if all executions under a plan agree, its execution residuals are zero.

For completeness, the importance ratio in Eq. (7) is

$$\rho_{k,m,t}(\theta) = \frac{\pi_\theta(z_{k,m,t} \mid x, z_{k,m,<t})}{\pi_{\text{old}}(z_{k,m,t} \mid x, z_{k,m,<t})}, \tag{15}$$

where $z_{k,m} = (c_k, y_{k,m})$. Separate span averages in Eq. (8) prevent plan credit from being diluted solely by a longer execution; they do not imply separate parameters or equal gradient norms. Plan reliability remains conditional on the rollout policy and decoding configuration, not an intrinsic proof label.

E.3 Plan-Token Contrast

For target plan c_k, the comparison set is $\mathcal{P}_k = \{j \neq k : c_j \text{ is valid}\}$. The highest-value index j_k^+ and lower-value set $\mathcal{N}_k$ follow Eq. (4). A valid contrast requires both. Higher and lower therefore denote empirical plan values rather than verified correctness of the plan text.

Each comparison plan is inserted into the same scoring template $H(x, c_j)$, and no execution is included. A frozen snapshot teacher-forces the target-plan tokens:

$$s_{k,t}(j) = \log \pi_{\text{snap}}(c_{k,t} \mid H(x, c_j), c_{k,<t}), \tag{16}$$

$$g_{k,t} = s_{k,t}(j_k^+) - \text{LME}_{j \in \mathcal{N}_k}\, s_{k,t}(j). \tag{17}$$

The target prefix $c_{k,<t}$ and template are identical across branches, so the difference isolates sensitivity to the comparison-plan content. These contexts are used only for scoring; the policy ratio still uses the original, unhinted rollout context. Snapshot scores and modulation are detached.

The bounded signal and activation mask are

$$d_{k,t} = \alpha \tanh(g_{k,t}/\tau_c), \qquad I_{k,t} = \mathbf{1}[\text{valid contrast and } |d_{k,t}| > \eta]. \tag{18}$$

Equation (6) is equivalent to clamping $A_k^c + I_{k,t} d_{k,t}$ to the half-line whose sign is fixed by A_k^c. Hence

$$A_k^c \widetilde{A}_{k,t}^c \geq 0, \qquad |\widetilde{A}_{k,t}^c - A_k^c| \leq \alpha. \tag{19}$$

Contrast can strengthen, weaken, or suppress plan-token credit, but cannot reverse verifier direction or change a zero plan advantage.

E.4 Degenerate Cases and Computation

With conditionally independent Bernoulli executions and $q_k = q_{\text{old}}(x, c_k)$,

$$\mathbb{E}[Q_k \mid x, c_k] = q_k, \qquad \text{Var}(Q_k \mid x, c_k) = \frac{q_k(1 - q_k)}{M}. \tag{20}$$

Increasing M improves precision for a fixed plan while reducing plan diversity $K = G/M$. With $M = 1$, execution residuals vanish; with $K = 1$, plan-relative credit vanishes. Target-excluding contrast requires at least three valid plans and differing values. Wherever no contrast can be formed, PC-GRPO reduces to PC-GRPO w/o contrastive reward.

Both variants use $G = KM$ verifier calls. PC-GRPO additionally scores one higher-value and $|\mathcal{N}_k|$ lower-value context for each valid target, but only over $|\mathcal{T}_{c_k}|$ plan tokens; these scores are shared across its M executions. Equal verifier counts therefore do not imply equal FLOPs or wall-clock time. At inference, the policy generates one plan and one execution without comparison sampling, snapshot scoring, or privileged information.

E.5 Prompt for Reasoning Outline Compression

We use the following prompt to compress detailed reasoning trajectories into concise high-level plans for constructing the training data.

Reasoning Outline Compression Prompt

Role
You are a high-level reasoning outline compressor.

Goal
Compress an existing detailed reasoning path into a very short high-level outline. The outline will be inserted between `<plan>` and `</plan>` in SFT data.

Core Principle
The outline should summarize groups of reasoning actions, not individual calculations. It may include key variables, numbers, equations, or constraints when they help define the strategy, but it must not become a step-by-step solution.

Important
The `reasoning_path` may contain wrong preliminary plans, false starts, self-corrections, repeated checks, or meta-reasoning. Ignore reasoning that is later corrected or contradicted. Keep only the final successful line of reasoning.

Compression Rules

1. Output exactly 2 or 3 numbered steps.
2. Each step should merge several low-level reasoning steps into one strategic action.
3. It is okay to mention important variables, given numbers, equations, or target quantities.
4. Avoid detailed algebra, arithmetic, simplification, factorization, or numerical substitutions.
5. Do not include the final answer unless it is explicitly needed to describe the strategy.
6. Do not describe verification, hesitation, or self-checking.
7. Do not copy the `reasoning_path` sentence by sentence.
8. Preserve the logical order of the final valid solution.
9. Prefer broad actions such as "model", "compare", "solve", "substitute", "evaluate", and "convert".
10. The outline should be concise but complete enough that a solver can reconstruct the full reasoning.

Problem
`{problem}`

Reasoning Path
`{reasoning_path}`

Task
Write a compressed high-level outline of the final successful reasoning.

Output Format
Use exactly 2 or 3 numbered steps. Each step must be one concise sentence. Do not wrap the output in `<plan>` tags.

E.6 Route-Guided Problem-Solving Prompt

We use this prompt to structure each rollout into a *plan-then-execute* process. The model first generates a concise high-level route within `<plan>` tags and then follows it to produce the detailed reasoning. The explicit plan boundary separates route formulation from subsequent execution during training.

Route-Guided Problem-Solving Prompt

Problem
{problem}

Solve the problem and put the final answer within \boxed{}.

Before detailed reasoning, write a compact high-level route scaffold inside <plan> and </plan>. The plan must satisfy all of the following:

1. Use exactly 2 or 3 numbered steps, with one concise strategic sentence per step.
2. Summarize groups of reasoning actions rather than individual calculations.
3. Name the key method, variables, equations, transformations, cases, or constraints when they define the route.
4. Preserve a logical solving order, but do not include detailed algebra, arithmetic, numerical substitutions, hesitation, or the final answer.
5. Make the plan concise but complete enough to guide the subsequent reasoning.

After </plan>, execute the plan, adapt when the mathematical evidence requires it, and finish with the boxed answer.