# METAOPSD: ADAPTING PRIVILEGED TEACHERS WITH POST-UPDATE OUTCOME FEEDBACK

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Access to reference solutions can make a language model a better predictor without making it a better teacher. On-policy self-distillation (OPSD) trains a problem-only student to match a reference-conditioned teacher on student-generated prefixes, but distributional agreement does not directly assess the learning gains induced by that supervision. We introduce **MetaOPSD**, which adapts the privileged teacher using verified student outcomes after a pilot distillation step. The pilot student solves separate feedback problems without privileged information, and a policy-gradient surrogate propagates outcome feedback through the pilot update to train the teacher. The adapted teacher then supervises an OPSD update from the original student state, retaining dense distillation as the student's training objective. MetaOPSD outperforms all evaluated baselines on each of three mathematical reasoning benchmarks at both model sizes after 200 student updates, with macro avg@12 gains of 2.16–2.96 percentage points over the OPSD baseline. On Qwen3-1.7B, checkpoint cross-evaluation further shows that the final teacher induces larger learning gains than the initial teacher at all evaluated student states.

## 1 INTRODUCTION

Privileged information can make a language model a better predictor without making it a better teacher. In on-policy self-distillation (OPSD), the student generates responses from the problem alone, while a teacher additionally sees a reference solution and provides dense next-token supervision on the student's own prefixes (Zhao et al., 2026). A KL-based distillation objective trains the student to match these predictions (Hinton et al., 2015). This allows the teacher to guide the student on the reasoning trajectories it actually encounters (Agarwal et al., 2024), with the goal of improving the student's ability to solve problems independently.

However, in our experience, OPSD can transfer output patterns from the privileged context to the student, even when they hinder problem solving. This observation is consistent with recent reports of reference-induced shortcuts and reasoning degradation (Shen et al., 2026; Kaur et al., 2026). The teacher can use the reference solution to predict the next step, while the student must learn to derive it from the problem alone. The KL-based objective encourages agreement with the teacher's predictions, but does not directly evaluate whether the resulting student update improves independent problem solving (Stanton et al., 2021).

We use *teaching utility* to measure how learning from a teacher changes the student's performance without privileged information. This utility depends on the student's current learning state: the same supervision can produce different gains at different stages of training (Zhu & Wang, 2021; Wu et al., 2018). We therefore use post-update student outcomes not only to evaluate the teacher, but also to adapt its supervision to the current student.

We introduce **MetaOPSD** (Figure 2), a meta-teaching algorithm which trains the teacher using the student's performance after a distillation step. Each cycle begins with a pilot OPSD update. The updated student then solves separate problems without privileged information. We construct a policy-gradient surrogate from its verified outcomes and differentiate it through the pilot update to train the teacher. The updated teacher then provides supervision for the actual student update, which remains an OPSD step. This makes the pilot student's problem-only performance the criterion for teacher adaptation. The teacher is thus trained to provide supervision that helps the student solve



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.

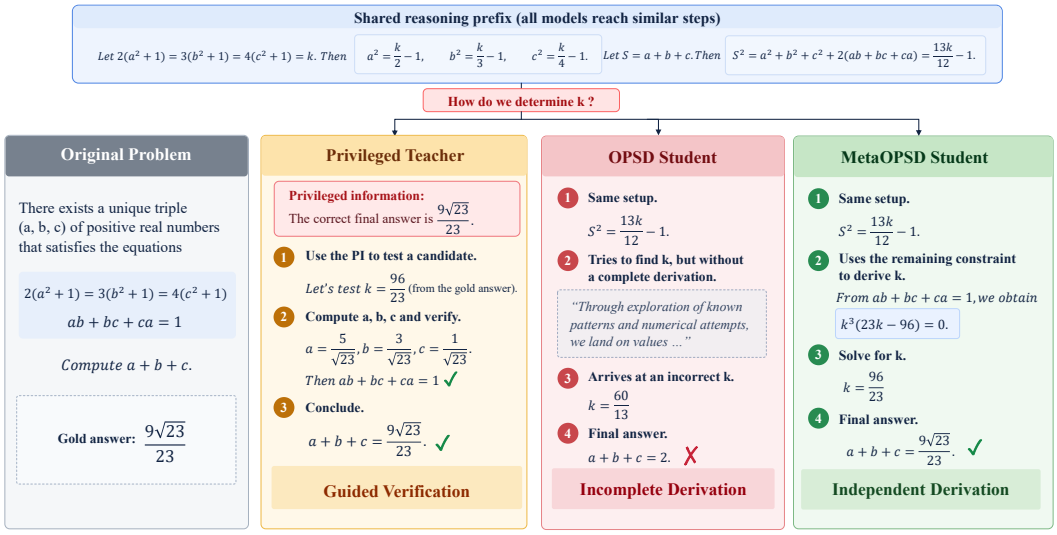


Figure 1: For the same problem, the privileged teacher uses the provided answer to identify and verify the correct intermediate value $k$, whereas the student must recover $k$ from the problem alone. The OPSD-trained student reaches the shared setup but fails at this bottleneck; the MetaOPSD-trained student derives the missing condition and reaches the correct answer without privileged information.

problems when privileged information is unavailable. Figure 1 illustrates this motivation through a comparison of responses to the same problem.

We evaluate MetaOPSD on Qwen3-1.7B and Qwen3-4B (Yang et al., 2025) across AIME24, AIME25, and HMMT25. After 200 student updates, MetaOPSD achieves the highest avg@12 among the evaluated methods on every benchmark at both model sizes. Its macro gains over OPSD are 2.96 and 2.16 percentage points, respectively, and its gains over VISTA (Ding et al., 2026), the strongest evaluated baseline by macro accuracy, are 1.98 and 1.36 points. To examine whether teacher adaptation improves teaching utility, we cross-evaluate teacher and student checkpoints on Qwen3-1.7B from matched student parameters and optimizer states. The final teacher induces larger learning gains than the initial teacher at all five evaluated student checkpoints. Replacing outcome feedback with reference negative log-likelihood also improves over frozen OPSD, but yields lower accuracy than MetaOPSD, supporting the value of verified post-update outcomes as feedback for teacher adaptation.

Our contributions are threefold. First, we formulate teaching utility for privileged on-policy self-distillation as the improvement in a student's problem-only performance after learning from a teacher, conditioned on the student's current learning state. Second, we introduce MetaOPSD, which differentiates a policy-gradient surrogate of verified post-update outcomes through a pilot distillation step to adapt the teacher while retaining OPSD as the student's training objective. Third, we show that MetaOPSD improves reasoning accuracy at both evaluated model sizes and, on Qwen3-1.7B, increases teaching utility relative to the initial teacher across five student states and outperforms adaptation based on reference likelihood.

## 2 Related Work

**Privileged supervision for on-policy distillation.** On-policy distillation trains on student-generated continuations (Agarwal et al., 2024; Gu et al., 2024). OPSD and SDPO enrich the teacher's context with reference solutions or feedback to provide dense supervision (Zhao et al., 2026; Hübotter et al., 2026). However, privileged targets can suppress self-correction or reduce distillation loss without improving task accuracy (Kaur et al., 2026; Harne et al., 2026). Purified OPSD removes reference-induced supervision components, DemoPSD blends teacher and student targets according to disagreement, and USD jointly controls exposure and token selection (Shen et al., 2026; Li et al., 2026; Yang et al., 2026b). MetaOPSD learns the teacher's supervision from its effect on student

learning, making independent problem-solving performance the criterion for adapting privileged targets.

**Adaptive privileged teachers.** DualOPSD fits the teacher to the updated student's distribution (Chen et al., 2026); PAST adapts it using complete student trajectories and verified outcomes (Feng et al., 2026). VISTA uses correct student rollouts to update the teacher at positions with large teacher–student disagreement (Ding et al., 2026). ATESD learns reference exposure from delayed distillation-loss progress and teacher likelihood on reference tokens (Han et al., 2026). MetaOPSD directly differentiates the updated student's correctness surrogate through the distillation step to train the teacher. This connects each teacher correction to how the current student learns and subsequently solves separate problems without privileged information. The resulting feedback targets the learning benefit of supervision, beyond matching student distributions, filtering successful rollouts, or adjusting reference exposure.

**Meta-teaching and outcome feedback.** Learning-to-teach methods adapt instruction to the learner (Fan et al., 2018; Wu et al., 2018). MetaDistil establishes a direct precedent for optimizing teachers through pilot student updates (Zhou et al., 2022). MetaOPSD extends this principle to privileged autoregressive reasoning, using verified, freely generated answers as post-update feedback. RLSD uses outcomes to guide student policy updates, while OGLS-SD calibrates teacher logits through successful–failed trajectory contrasts (Yang et al., 2026a;c). In MetaOPSD, an RLOO surrogate (Ahmadian et al., 2024) instead propagates outcome feedback through the student's learning step to the teacher. The committed student update remains OPSD, allowing sparse correctness rewards to improve dense token-level supervision while preserving the student's distillation objective.

## 3 METAOPSD

We study on-policy self-distillation with reference solutions available during training (Zhao et al., 2026). Each training example consists of a problem $x$, a reference solution $z$, and a verifier target $a$. The student $p_\theta$ generates a response $y$ from $x$ alone, while the privileged teacher $q_\phi$ additionally receives $z$ to provide next-token supervision. An external verifier uses $a$ to assign a binary correctness reward $r(y, a) \in \{0, 1\}$ to the student's response.

MetaOPSD trains the teacher using the student's performance feedback, which is collected after learning from its supervision. Each cycle begins with a pilot OPSD step on a teaching batch $\mathcal{B}_{\text{teach}}$, producing temporary student parameters while preserving the incoming student and optimizer state. This pilot student then generates responses on a feedback batch $\mathcal{B}_{\text{meta}}$. The verified outcomes provide a policy-gradient estimate, which is propagated through the pilot update to train the teacher. The updated teacher then supervises one OPSD step from the original student parameters and optimizer state. Both batches are drawn from the training pool, with disjoint problem groups within each cycle.

Figure 2 illustrates the interaction between student learning and teacher adaptation. Algorithm 1 summarizes one training cycle.

### 3.1 OPSD WITH SEPARATE STUDENT AND TEACHER ADAPTERS

The student and privileged teacher share a frozen backbone with separate trainable adapters: student LoRA parameters $\theta$ (Hu et al., 2022) and teacher logit-adapter parameters $\phi$. The student is updated through OPSD in the inner loop, while the teacher is updated from the resulting student's outcomes in the outer loop. The teacher is parameterized as

$$\ell_\phi(x, z, y_{<j}) = \ell_0(x, z, y_{<j}) + B_\phi A_\phi h_0(x, z, y_{<j}). \tag{1}$$

Here $h_0$ and $\ell_0$ are the frozen backbone's hidden representation and logits. The trainable matrices $A_\phi$ and $B_\phi$ provide a low-rank correction to these logits.

For each teaching problem, the student generates a response from the problem alone. The teacher additionally receives the reference solution and supervises the same response prefixes through the OPSD objective (Zhao et al., 2026):

$$\mathcal{L}_{\text{teach}}(\theta, \phi) = \mathbb{E}_{x,z,y_{<j}} \left[ D(q_\phi(\cdot \mid x, z, y_{<j}) \,\|\, p_\theta(\cdot \mid x, y_{<j})) \right]. \tag{2}$$

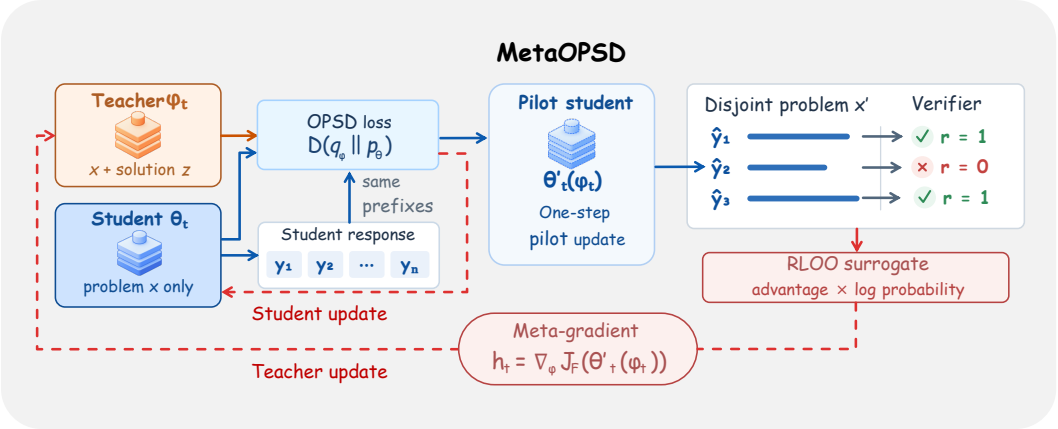


Figure 2: **Overview of MetaOPSD.** We first take one OPSD step on a pilot student using supervision from the current teacher. The pilot student answers separate feedback problems, and verified outcomes define an RLOO loss. Differentiating this loss through the pilot update trains the teacher. The updated teacher then supervises one OPSD step from the original student parameters and optimizer state.

The expectation averages over valid response prefixes generated by the incoming student on the teaching batch. $D$ denotes the forward-KL-based distillation discrepancy; implementation details are given in Appendix A.

The inner loop holds the teacher adapter fixed and updates the student adapter through this objective. This pilot step produces $\theta'_t(\phi)$, whose post-update outcomes provide feedback for training the teacher adapter in the outer loop.

### 3.2 Outcome feedback from the pilot student

To obtain feedback for teacher adaptation, we evaluate the pilot student $\theta'_t(\phi_t)$ after its OPSD update. On a feedback batch $\mathcal{B}_{\text{meta}}$ of $B$ problems, disjoint from the teaching problems within the current cycle, the pilot student independently generates $K \geq 2$ responses per problem using only the problem statement. An external verifier assigns a binary correctness reward $r_{b,k}$ to each response $\widehat{y}_{b,k}$.

We use REINFORCE with a leave-one-out baseline (RLOO) (Ahmadian et al., 2024) to obtain a policy-gradient estimate from the verified outcomes. For each response, the advantage is

$$A_{b,k} = r_{b,k} - \frac{1}{K-1} \sum_{\ell \neq k} r_{b,\ell}. \tag{3}$$

These advantages define the policy-gradient surrogate:

$$J_F(\theta') = -\frac{1}{BK} \sum_{b=1}^{B} \sum_{k=1}^{K} A_{b,k} \log p_{\theta'}(\widehat{y}_{b,k} \mid x_b). \tag{4}$$

Here $\theta'$ denotes the student parameters used to score the collected responses, and each response log-probability is summed over its generated tokens. Responses and advantages remain fixed during differentiation.

This surrogate provides a gradient signal favoring responses with higher relative correctness. The next section uses this signal to train the teacher by differentiating through the pilot student update. Feedback collection details are given in Appendix A.

### 3.3 Teacher adaptation through the student update

The outer loop trains the teacher using the outcome feedback of the student it induces. With the collected responses and advantages held fixed, the teacher objective is

$$F_t(\phi) = J_F(\theta'_t(\phi)) \,. \tag{5}$$

**Algorithm 1: One MetaOPSD training cycle**

**Input:** Student $\theta_t$, teacher $\phi_t$; disjoint batches $\mathcal{B}_{\text{teach}}$ and $\mathcal{B}_{\text{meta}}$.

1. Sample teaching responses $y_i \sim p_{\theta_t}(\cdot \mid x_i)$ on $\mathcal{B}_{\text{teach}}$.
2. Update a copy of $\theta_t$ by OPSD with $\phi_t$ to obtain the pilot $\theta'_t(\phi_t)$.
3. Generate $K$ responses per problem on $\mathcal{B}_{\text{meta}}$ using the pilot student.
4. Verify the responses and compute the RLOO loss $J_F$.
5. Update $\phi_t$ to $\phi_{t+1}$ using the meta-gradient $h_t$ through the pilot update.
6. Update the original student with $\phi_{t+1}$: $\theta_{t+1} \leftarrow \theta'_t(\phi_{t+1})$.

Both OPSD steps use the same teaching responses and incoming student optimizer state.

Differentiating through the pilot OPSD update (Franceschi et al., 2018; Maclaurin et al., 2015) gives

$$h_t = \nabla_\phi F_t(\phi_t) = \left(\frac{\partial \theta'_t}{\partial \phi}\right)^\top \nabla_{\theta'} J_F(\theta'), \tag{6}$$

where the derivatives are evaluated at $\phi = \phi_t$ and $\theta' = \theta'_t(\phi_t)$.

The policy gradient with respect to the pilot student is propagated through the OPSD update to obtain the teacher meta-gradient. We differentiate through only the current student step, holding the incoming student parameters and optimizer state fixed.

Under independent on-policy sampling of complete response groups and a locally differentiable student update, $h_t$ estimates the negative gradient of the updated student's expected correctness on the feedback problems with respect to the teacher parameters, evaluated at $\phi_t$. Appendix B gives the derivation.

We use $h_t$ to update the teacher parameters, obtaining $\phi_{t+1}$. Appendix A specifies the teacher optimizer and regularization. The updated teacher then provides supervision on the same teaching responses for the student update:

$$\theta_{t+1} = \theta'_t(\phi_{t+1}). \tag{7}$$

This update starts from the original student parameters $\theta_t$ and optimizer state $S_t$. The pilot update supplies the teacher's learning signal, while the updated teacher determines the actual OPSD step. The resulting student parameters and optimizer state are retained for the next cycle. Each completed cycle advances the student by exactly one OPSD step, with outcome feedback training the teacher that supplies its supervision.

## 4 Experiments

We test whether MetaOPSD improves student reasoning, whether adapted teachers induce larger learning gains, and how outcome feedback compares with reference likelihood.

### 4.1 Experimental setup

**Training.** We train Qwen3-1.7B and Qwen3-4B (Yang et al., 2025) on up to 30K problem–solution pairs from the mathematical reasoning subset of OpenThoughts (Guha et al., 2025). MetaOPSD uses a shared frozen backbone, rank-64 student LoRA (Hu et al., 2022), and a separate rank-64 teacher logit adapter. Each MetaOPSD cycle uses 32 teaching problems and four disjoint feedback problems, with four responses sampled per feedback problem. Within each model size, all trained methods start from the same student checkpoint and optimizer state, use the same problem pool and rank-64 student LoRA ($\alpha = 128$), and receive 200 student updates in the main comparison. We evaluate every 40 updates and report the step-200 endpoint in benchmark tables. Table A.1 gives the training configuration.

**Comparisons.** **Base** uses the pretrained model without task-specific training. **SFT** fits OpenThoughts solutions from problem-only prompts; **GRPO** (Shao et al., 2024) optimizes sampled responses with the same binary correctness verifier as MetaOPSD. **OPSD** (Zhao et al., 2026) freezes

Table 1: **Mathematical reasoning performance (avg@12, %).** Trained methods are evaluated at the fixed student step 200; all scores use three-seed aggregation. Compute and feedback budgets are method-specific. Macro is the unweighted mean across benchmarks; averages and gains are computed before rounding. Bold marks column maxima within each model group; each group's final row gives gains over OPSD in percentage points.

| Method | AIME24 | AIME25 | HMMT25 | Macro Avg. |
|---|---|---|---|---|
| **Qwen3-1.7B** | | | | |
| Base | 47.78 | 36.39 | 25.00 | 36.39 |
| SFT | 49.44 | 37.22 | 25.56 | 37.41 |
| GRPO | 48.33 | 37.50 | 25.28 | 37.04 |
| OPSD | 51.30 | 38.06 | 28.06 | 39.14 |
| DualOPSD | 49.91 | 37.31 | 26.20 | 37.81 |
| VISTA | 52.31 | 38.61 | 29.44 | 40.12 |
| **MetaOPSD** | **55.19** | **39.81** | **31.30** | **42.10** |
| MetaOPSD − OPSD | +3.89 | +1.76 | +3.24 | +2.96 |
| **Qwen3-4B** | | | | |
| Base | 74.72 | 65.56 | 42.96 | 61.08 |
| SFT | 75.74 | 65.93 | 43.70 | 61.79 |
| GRPO | 75.56 | 67.22 | 44.54 | 62.44 |
| OPSD | 76.30 | 68.06 | 45.37 | 63.24 |
| DualOPSD | 76.94 | 68.70 | 46.02 | 63.89 |
| VISTA | 77.04 | 68.98 | 46.11 | 64.04 |
| **MetaOPSD** | **78.24** | **70.37** | **47.59** | **65.40** |
| MetaOPSD − OPSD | +1.94 | +2.31 | +2.22 | +2.16 |

the initial privileged teacher and uses the same clipped student distillation objective as MetaOPSD. **DualOPSD** (Chen et al., 2026) and **VISTA** (Ding et al., 2026) retain OPSD student training while adapting the teacher through distribution matching and verifier feedback, respectively. This comparison fixes student updates; teacher adaptation and feedback computation remain method-specific. Section 4.5 reports training costs and resource-matched comparisons; Appendix D.4 gives additional training settings.

**Evaluation.** We evaluate on AIME 2024 (AIME24), AIME 2025 (AIME25), and HMMT February 2025 (HMMT25), each containing 30 problems. We generate 12 problem-only responses per question in thinking mode and report avg@12, the mean fraction of correct responses per problem. Macro accuracy averages the three benchmarks equally. Scores are percentages averaged over three random seeds; differences are percentage points.

### 4.2 Reasoning performance

**Reasoning gains.** At step 200, MetaOPSD leads all three benchmarks at both model sizes (Table 1). Its Macro accuracies of 42.10% / 65.40% on 1.7B / 4B exceed OPSD by 2.96 / 2.16 points and VISTA, the strongest Macro baseline, by 1.98 / 1.36 points. DualOPSD trails OPSD on 1.7B and exceeds it on 4B.

**Learning curves.** Figure 3 shows that MetaOPSD attains the highest Macro accuracy from step 80 onward at both model sizes. Its advantage over VISTA grows from 0.09 to 1.98 points on 1.7B and from 0.34 to 1.36 points on 4B between steps 80 and 200.

**Teacher-update frequency.** On Qwen3-1.7B, teacher-update intervals $k \in \{1, 5, 10, 20\}$ yield step-200 Macro accuracies of 42.10%, 40.80%, 39.66%, and 38.09%, with the same ordering across

benchmarks (Figure 4). The sweep changes both adaptation frequency and feedback use: $k = 1$ performs 200 adaptation cycles, versus 10 at $k = 20$.

### 4.3 TEACHING UTILITY

**Checkpoint cross-evaluation.** On Qwen3-1.7B, each seed's MetaOPSD run supplies five student checkpoints, $\mathcal{S} = \{40, 80, 120, 160, 200\}$, with six teacher checkpoints, $t \in \{0, 40, 80, 120, 160, 200\}$. For each student, six branches share parameters, AdamW (Loshchilov & Hutter, 2019) state, teaching problems, and update settings; each learns from one frozen teacher for $H = 20$ on-policy OPSD updates. We evaluate the starting student and endpoints on 128 MATH-500 questions (Hendrycks et al., 2021; Lightman et al., 2023), excluded from training, feedback, development, and final-benchmark data, with 12 problem-only responses per question and three-seed averaging.

Writing $\widehat{V}_P$ for mean probe accuracy in percent, teaching utility is

$$U_{s,t} = \widehat{V}_P\left(\theta_s^{(H)}(q_t)\right) - \widehat{V}_P(\theta_s), \tag{8}$$

where $\theta_s^{(H)}(q_t)$ is the student after learning from $q_t$. The mean advantage over $q_0$ on the fixed student set is

$$\overline{\Delta}(t) = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} (U_{s,t} - U_{s,0}) \,. \tag{9}$$

**Improved teaching utility.** The final teacher $q_{200}$ exceeds $q_0$ in utility by 0.41–0.54 points across all five student checkpoints (Figure 5(a)). The mean advantage over $q_0$ rises from 0.16 points at teacher step 40 to 0.48 points at step 200 (Figure 5(b)). The final teacher has the highest mean utility, while individual students can favor earlier teachers ($s = 120$ favors $q_{160}$).

**Variation across student stages.** Later students generally gain less from additional distillation. For $q_{200}$, teaching utility is 0.72 percentage points at student step 40 and 0.02 points at student step 200. At this final student state, $q_0$ reduces probe accuracy by 0.39 points: teacher adaptation improves the supervision even when further learning gains are small.

### 4.4 REFERENCE NLL AS META-FEEDBACK

We replace outcome feedback with reference negative log-likelihood (NLL) to test whether fitting reference solutions provides equally effective feedback for teacher adaptation.

**Reference-based meta-feedback.** For feedback problem $x_b$ and reference solution $z_b$ with $L_b$ scored tokens, the alternative objective is

$$J_{\mathrm{ref}}(\theta') = -\frac{1}{B} \sum_{b=1}^{B} \frac{1}{L_b} \sum_{j=1}^{L_b} \log p_{\theta'}(z_{b,j} \mid x_b, z_{b,<j}). \tag{10}$$

The reference includes reasoning and the final answer; token losses are averaged within each problem, then equally across problems. We substitute $J_{\mathrm{ref}}$ for $J_F$ in Equation (6), retaining differentiation through the pilot update and OPSD training for the real student.

**Matched student training.** Frozen OPSD, Reference NLL, and MetaOPSD share the student initialization, optimizer state, teaching-problem stream, and 200-update budget. The adaptive variants also share teacher settings and feedback problems; NLL scores references in place of sampling feedback responses. Appendix D.3 gives the full protocol and scoring details.

**Outcome feedback yields stronger reasoning.** Table 2 shows that Reference NLL reaches 41.23% Macro accuracy and improves over frozen OPSD on all three benchmarks, with a Macro gain of +2.10 points. MetaOPSD further improves over Reference NLL by 0.86 points, reaching 42.10%. Its largest gain is on AIME25 (1.57 points). Both post-update objectives improve on frozen supervision, with outcome feedback yielding stronger problem-only reasoning.

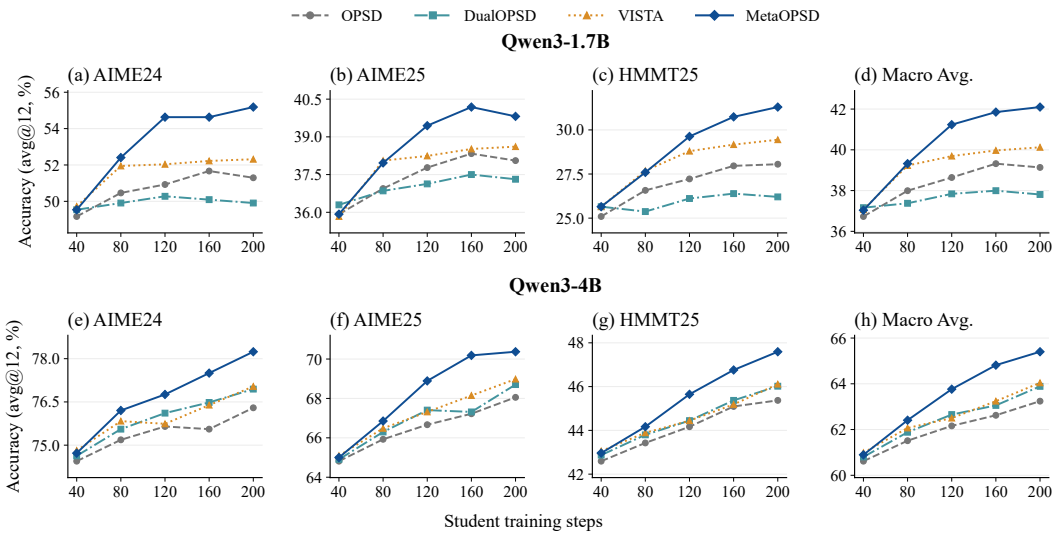


Figure 3: **Reasoning learning curves.** Avg@12 over student updates for Qwen3-1.7B (top row) and Qwen3-4B (bottom row). Columns show AIME24, AIME25, HMMT25, and their unweighted Macro average.

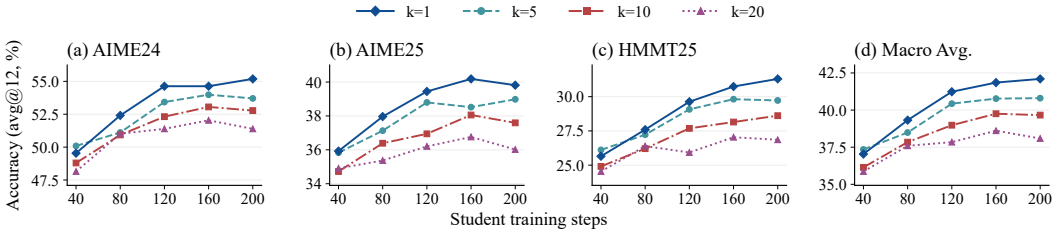


Figure 4: **Effect of teacher-update frequency on reasoning performance.** Qwen3-1.7B avg@12 with teacher adaptation every $k \in \{1, 5, 10, 20\}$ student steps and a fixed budget of 200 student updates. Standard MetaOPSD uses $k = 1$; intervening steps use OPSD with the current teacher. Only supplied means are shown.

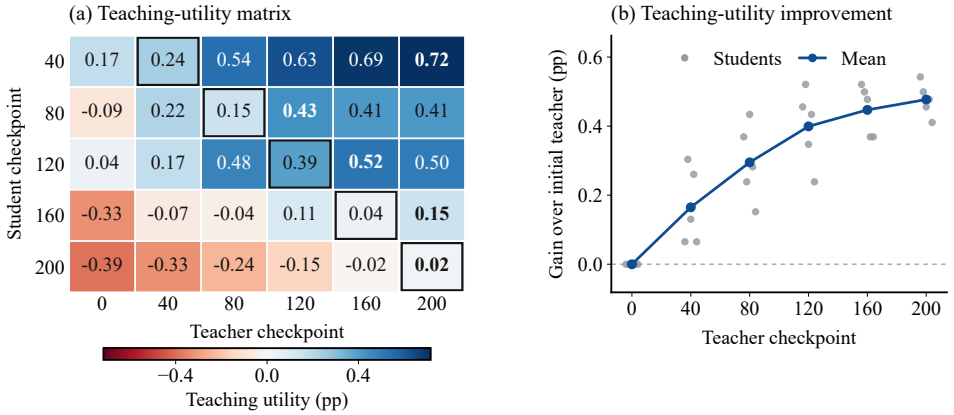


Figure 5: **Teacher–student checkpoint cross-evaluation.** (a) Teaching utility $U_{s,t}$ in percentage points. Outlined cells indicate stage-matched teacher–student pairs; bold entries mark row maxima. (b) Gain over the initial teacher $q_0$, shown for individual student checkpoints (gray points) and their mean $\overline{\Delta}(t)$ (blue line).

Table 2: **Effect of the meta-feedback objective on Qwen3-1.7B.** Avg@12 (%) at student step 200, with three-seed aggregation. Macro is the unweighted mean across benchmarks; averages and gains are computed before rounding. Reference NLL and MetaOPSD adapt the teacher through a pilot student update using reference likelihood and outcome feedback, respectively; OPSD keeps the teacher frozen.

| Teacher setting | AIME24 | AIME25 | HMMT25 | Macro Avg. |
|---|---|---|---|---|
| Frozen OPSD | 51.30 | 38.06 | 28.06 | 39.14 |
| Reference NLL | 54.63 | 38.24 | 30.83 | 41.23 |
| MetaOPSD | 55.19 | 39.81 | 31.30 | 42.10 |

### 4.5 Training efficiency

Table 3 compares OPSD, VISTA, and MetaOPSD with RLOO or reference NLL (Section 4.4) for Qwen3-1.7B on 8 A6000 48GB GPUs. The shared budget is 24 GPU-hours. Each method uses its last complete checkpoint within budget, allowing different student-step counts. Training time covers the complete cycle, including feedback computation and teacher adaptation; evaluation is timed separately. For 200 updates, RLOO takes 24.994 hours, compared with 7.095 for NLL. NLL scores fixed reference tokens in parallel; RLOO adds autoregressive generation and verification of four responses per feedback problem. MetaOPSD (RLOO) reaches the OPSD target in 9.998 hours, versus 2.767 for OPSD. At this budget, its Macro differences from OPSD and VISTA are -0.20 and -1.62 points, respectively.

Table 3: **Training cost and student quality on Qwen3-1.7B.** $T_{200}$ and $G_{200}$ are hours and GPU-hours for 200 student updates; $T_{\text{match}}$ is the earliest evaluated time reaching the paired OPSD run's step-200 Macro. $A_G$ is Macro avg@12 (%) at a shared GPU-hour budget. Entries average three runs.

| Method | $T_{200}$ | $G_{200}$ | $T_{\text{match}}$ | $A_G$ |
|---|---|---|---|---|
| OPSD | 3.459 | 27.672 | 2.767 | 38.24 |
| VISTA | 4.228 | 33.824 | 2.091 | 39.66 |
| MetaOPSD (NLL) | 7.095 | 56.760 | 4.257 | 38.31 |
| MetaOPSD (RLOO) | 24.994 | 199.952 | 9.998 | 38.04 |

## 5 Limitations and Discussion

Like bilevel meta-learning methods such as MetaDistil (Zhou et al., 2022), MetaOPSD performs two coupled update stages for the teacher and student. Differentiating through the pilot student step raises the cost per training cycle, while allowing the teacher to adapt its supervision to how the student learns. Our experiments show higher final student accuracy at the fixed update count. The fixed-budget and time-to-target comparisons favor cheaper baselines (Section 4.5). Reference NLL offers a less expensive adaptive variant; RLOO incurs additional cost to learn from the outcomes of the student's own responses. More economical feedback collection could improve this tradeoff. The additional computation is confined to training: deployment uses the same student architecture without the teacher or verifier. Thus, whether the added training cost is worthwhile depends on the deployment scale and the value placed on improved student quality.

## 6 Conclusion

MetaOPSD adapts privileged supervision through verified student outcomes after a pilot distillation step, retaining OPSD as the student's objective. It improves mathematical reasoning at both model sizes after 200 student updates. On Qwen3-1.7B, checkpoint cross-evaluation shows higher teaching utility for the final teacher across the five tested student states. Together, these results support optimizing privileged supervision for the student's ability to solve problems independently.

## Reproducibility Statement

Section 3 and Algorithm 1 describe the MetaOPSD training procedure. Appendix A specifies the student update, teacher adaptation, and optimizer-state handling, while Appendix B derives the outcome-based teacher gradient. Table A.1 reports the training configuration. Appendix D documents the teaching-utility probe, baseline settings, and reference-likelihood comparisons.

## Ethics Statement

This work studies language-model training for mathematical reasoning using existing models, problem–solution datasets, and automated answer verification. Relevant ethical considerations include the provenance and permitted reuse of training data and the reliability of generated solutions. Use and redistribution of third-party datasets and model weights are subject to their respective licenses and terms. Models trained with this method can still produce incorrect or misleading reasoning; downstream applications require validation appropriate to their intended use.

## AI Use Statement

Generative AI tools assisted with experimental design and equation drafting. They also supported manuscript organization and prose revision, related-work searches, and LaTeX editing and compilation. The authors take responsibility for the final content of the paper, including its methodology, references, claims, reported results, and AI-assisted content.

## References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2306.13649.

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, 2024. URL https://arxiv.org/abs/2402.14740.

Yutong Chen, Guangfu Guo, Zhichao Xu, and Kunpeng Liu. DualOPSD: Adaptive privileged teachers for on-policy self-distillation, 2026. URL https://arxiv.org/abs/2608.26019.

Zewen Ding, Zezhong Wu, Zhou Tao, Shida Wang, Shizhuo Hou, YongXiang Hua, Haoyu Cao, and Linli Xu. VISTA: Verifier-informed student-to-teacher adaptation for on-policy self-distillation, 2026. URL https://arxiv.org/abs/2608.28306.

Yang Fan, Fei Tian, Tao Qin, Xiang-Yang Li, and Tie-Yan Liu. Learning to teach. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HJewuJWCZ.

Yangyang Feng, Zhuoyan Feng, and Junlan Chen. PAST: Privileged adaptation from complete student trajectories for on-policy self-distillation, 2026. URL https://arxiv.org/abs/2608.08726.

Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1568–1577. PMLR, 2018. URL https://proceedings.mlr.press/v80/franceschi18a.html.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2306.08543v4.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, et al. OpenThoughts: Data recipes for reasoning models, 2025. URL https://arxiv.org/abs/2506.04178.

Zihao Han, Tiangang Zhang, Huaibin Wang, and Yilun Sun. Adaptive teacher exposure for self-distillation in LLM reasoning, 2026. URL https://arxiv.org/abs/2605.11458.

Sarthak Harne, Chinmay Karkar, Yash Pandya, Ahmed Awadallah, and Akshay Nambi. Privileged, but biased: How PI-conditioned teachers break self-distillation, 2026. URL https://arxiv.org/abs/2608.04794.

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 *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://arxiv.org/abs/2103.03874.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. URL https://arxiv.org/abs/1503.02531.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, and Andreas Krause. Reinforcement learning via self-distillation, 2026. URL https://arxiv.org/abs/2601.20802.

Simran Kaur, Narutatsu Ri, Yinghui He, Liam Fowl, and Sanjeev Arora. Rethinking on-policy self-distillation for thinking models, 2026. URL https://arxiv.org/abs/2607.05184.

Yunhe Li, Hao Shi, Wenhao Liu, Mengzhe Ruan, Hanxu Hou, Zhongxiang Dai, Shuang Qiu, and Linqi Song. DemoPSD: Disagreement-modulated policy self-distillation, 2026. URL https://arxiv.org/abs/2607.02502.

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, 2023. URL https://arxiv.org/abs/2305.20050.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Dougal Maclaurin, David Duvenaud, and Ryan Adams. Gradient-based hyperparameter optimization through reversible learning. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2113–2122. PMLR, 2015. URL https://proceedings.mlr.press/v37/maclaurin15.html.

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.

Zhanming Shen, Jintao Tong, Shaotian Yan, Chen Shen, Hao Chen, Wentao Ye, Xiaomeng Hu, Rui Miao, Haobo Wang, Junbo Zhao, Gang Chen, and Jieping Ye. Purified OPSD: On-policy self-distillation without losing how to think, 2026. URL https://arxiv.org/abs/2607.02234.

Samuel Stanton, Pavel Izmailov, Polina Kirichenko, Alexander A. Alemi, and Andrew G. Wilson. Does knowledge distillation really work? In *Advances in Neural Information Processing Systems*, volume 34, pp. 6906–6919. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/hash/376c6b9ff3bedbbea56751a84fffc10c-Abstract.html.

Lijun Wu, Fei Tian, Yingce Xia, Yang Fan, Tao Qin, Jian-Huang Lai, and Tie-Yan Liu. Learning to teach with dynamic loss functions. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/hash/8051a3c40561002834e59d566b7430cf-Abstract.html.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Chenxu Yang, Chuanyu Qin, Qingyi Si, Minghui Chen, Naibin Gu, Dingyu Yao, Zheng Lin, Weiping Wang, Jiaqi Wang, and Nan Duan. Self-distilled RLVR, 2026a. URL https://arxiv.org/abs/2604.03128.

Yongkang Yang, Zhezheng Hao, Hong Zhang, Yi Liu, Xiankun Lin, Wence Ji, Fanjunduo Wei, Jiarui Yu, Qiang Lin, Xiaoyun Liang, and Hande Dong. Matching supervision to the student's learning capacity: A unified framework for on-policy self-distillation, 2026b. URL https://arxiv.org/abs/2608.08176.

Yuxiao Yang, Xiaoyun Wang, and Weitong Zhang. OGLS-SD: On-policy self-distillation with outcome-guided logit steering for LLM reasoning, 2026c. URL https://arxiv.org/abs/2605.12400.

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, 2026. URL https://arxiv.org/abs/2601.18734.

Wangchunshu Zhou, Canwen Xu, and Julian McAuley. BERT learns to teach: Knowledge distillation with meta learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7037–7049, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.485. URL https://aclanthology.org/2022.acl-long.485/.

Yichen Zhu and Yi Wang. Student customized knowledge distillation: Bridging the gap between student and teacher. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5057–5066, October 2021. URL https://openaccess.thecvf.com/content/ICCV2021/html/Zhu_Student_Customized_Knowledge_Distillation_Bridging_the_Gap_Between_Student_and_ICCV_2021_paper.html.

# A Distillation and One-Step Updates

This appendix specifies the update maps in Section 3. Throughout one cycle, the incoming student parameters $\theta_t$, optimizer state $S_t$, and sampled teaching responses are fixed inputs. The pilot uses $\phi_t$ to obtain outcome feedback; the actual student step uses $\phi_{t+1}$ and starts from the same $(\theta_t, S_t)$.

## A.1 Teacher parameterization and distillation loss

For a teaching position $(i, j)$, write $s_{\theta,ij}$ for the student's logits given $(x_i, y_{i,<j})$. The teacher receives $(x_i, z_i, y_{i,<j})$ and uses

$$\begin{aligned} \ell_{\phi,ij} &= \ell_{0,ij} + B_\phi A_\phi h_{0,ij}, \\ q^{(T)}_{\phi,ij} &= \mathrm{softmax}(\ell_{\phi,ij}/T), \qquad p^{(T)}_{\theta,ij} = \mathrm{softmax}(s_{\theta,ij}/T). \end{aligned} \tag{A.1}$$

The frozen features $h_{0,ij}$ and logits $\ell_{0,ij}$ are computed with the student adapter disabled. For hidden width $d$, vocabulary $\mathcal{V}$, and teacher rank $r_\phi$, $A_\phi \in \mathbb{R}^{r_\phi \times d}$ and $B_\phi \in \mathbb{R}^{|\mathcal{V}| \times r_\phi}$. The experiments use $r_\phi = 64$, random initialization of $A_\phi$, and $B_\phi = 0$, so the initial teacher equals the frozen privileged distribution $q^{(T)}_0$.

Let $\mathcal{I}_T$ contain the valid response positions across the teaching batch, and let $N_T = |\mathcal{I}_T|$. The discrepancy $D$ in Equation (2) gives the empirical loss

$$\begin{aligned} d_{ijv}(\theta, \phi) &= q^{(T)}_{\phi,ij}(v) \left[ \log q^{(T)}_{\phi,ij}(v) - \log p^{(T)}_{\theta,ij}(v) \right], \\ \mathcal{L}_{\mathrm{teach}}(\theta, \phi) &= \frac{1}{N_T} \sum_{(i,j) \in \mathcal{I}_T} \sum_{v \in \mathcal{V}} \min\{d_{ijv}(\theta, \phi), c\}. \end{aligned} \tag{A.2}$$

The current configuration uses $T = 1.0$ and $c = 0.05$. The vocabulary is summed before averaging over valid response tokens, with no additional $T^2$ factor. Prompt and padding positions are excluded from $\mathcal{I}_T$. Teaching responses are generated once by $p_{\theta_t}$ and held fixed for both student updates in the cycle.

**Numerical evaluation.** The teacher residual, softmax, and vocabulary-term calculations use FP32; the shared backbone and student use BF16. The distillation loss is evaluated in chunks of 64 response positions, retaining the full vocabulary at each position. Each chunk uses the same global token denominator $N_T$. The teacher and student prompts are given in Appendix C.2.

**Student logit gradient.** At one position, abbreviate the distributions by $q_v, p_v$, and let $L_{ij} = \sum_v \min\{d_v, c\}$. On a neighborhood with no $d_v = c$, define

$$\chi_v = \mathbf{1}\{d_v < c\}, \qquad M = \sum_v \chi_v q_v. \tag{A.3}$$

Since $\partial \log p_v / \partial s_u = (\mathbf{1}\{v = u\} - p_u)/T$, differentiating the active terms yields

$$\frac{\partial L_{ij}}{\partial s_u} = -\frac{1}{T} \sum_v \chi_v q_v (\mathbf{1}\{v = u\} - p_u) = \frac{M p_u - \chi_u q_u}{T}. \tag{A.4}$$

When all terms are active, $M = 1$ and this becomes $(p_u - q_u)/T$. The factor $M p_u$ retains the coupling among logits through softmax.

**Dependence on the teacher.** At the fixed student state, $p_u$ and the student's logit Jacobian are independent of $\phi$. On the same active branch,

$$\begin{aligned} \nabla_\phi \frac{\partial L_{ij}}{\partial s_u} &= \frac{1}{T} \left( p_u \sum_v \chi_v \nabla_\phi q_v - \chi_u \nabla_\phi q_u \right), \\ \frac{\partial q_v}{\partial \ell_{\phi,u}} &= \frac{q_v}{T} (\mathbf{1}\{v = u\} - q_u). \end{aligned} \tag{A.5}$$

Writing $J_{ij} = \partial s_{\theta,ij}/\partial\theta$ at $\theta_t$ and $a_{ij} = \nabla_{s_{\theta,ij}} L_{ij}$ gives

$$g_t(\phi) = \frac{1}{N_T} \sum_{(i,j)\in\mathcal{I}_T} J_{ij}^\top a_{ij}, \qquad \frac{\partial g_t}{\partial \phi} = \frac{1}{N_T} \sum_{(i,j)\in\mathcal{I}_T} J_{ij}^\top \frac{\partial a_{ij}}{\partial \phi}. \tag{A.6}$$

Thus the student gradient is taken with respect to $\theta$ while retaining its dependence on the teacher parameters.

### A.2 Functional student update with the incoming optimizer state

The student map used by both the pilot and the actual update is

$$\begin{aligned} g_t(\phi) &= \nabla_\theta \mathcal{L}_{\text{teach}}(\theta, \phi)|_{\theta=\theta_t}, \\ (\theta_t'(\phi), S_t'(\phi)) &= \mathcal{U}(\theta_t, S_t, g_t(\phi)). \end{aligned} \tag{A.7}$$

First concatenate the trainable student gradients and apply global norm clipping:

$$n = \|g\|_2, \qquad \rho(g) = \min\left\{1, \frac{\tau}{n+\epsilon_c}\right\}, \qquad \bar{g} = C(g) = \rho(g) g, \tag{A.8}$$

where $\tau = 0.1$ and $\epsilon_c = 10^{-6}$. Gradients are aggregated across devices before computing this norm.

For one AdamW parameter group (Loshchilov & Hutter, 2019), let $m, v$ be the incoming moments and $s$ the number of completed updates. With $u = \bar{g}$, the functional step is

$$\begin{aligned} m^+ &= \beta_1 m + (1-\beta_1)u, & v^+ &= \beta_2 v + (1-\beta_2) u \odot u, \\ b_1 &= 1 - \beta_1^{s+1}, & b_2 &= 1 - \beta_2^{s+1}, \\ \widehat{m} &= m^+/b_1, & \widehat{v} &= v^+/b_2, \\ \theta' &= (1-\eta_\theta w)\theta_t - \eta_\theta \widehat{m} \oslash (\sqrt{\widehat{v}} + \epsilon_a). \end{aligned} \tag{A.9}$$

Here $\odot$ and $\oslash$ denote elementwise multiplication and division. The returned state contains $m^+, v^+, s+1$ and the optimizer group settings. Each parameter uses its own stored step counter; $s$ suppresses that indexing. The configuration uses $\eta_\theta = 5 \times 10^{-6}$, $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon_a = 10^{-8}$, and $w = 0$. The incoming moments and counters are held fixed during differentiation, while the dependence of both updated moments on $u$ is retained.

### A.3 Fixed feedback trajectories and RLOO normalization

The pilot $\theta_t'(\phi_t)$ generates $K$ independent responses to each of $B$ feedback problems. The experiments use $B = 4$ and $K = 4$. Feedback decoding uses temperature one, top-$p$ one, no top-$k$ filtering, and a 4,096-token cap. The student receives the problem alone, and the verifier supplies $r_{b,k} \in \{0, 1\}$. Let $L_{b,k}$ be the number of generated tokens scored under the generation stopping rule. With the complete response groups in Equation (3), the surrogate is

$$J_F(\vartheta) = -\frac{1}{BK} \sum_{b=1}^{B} \sum_{k=1}^{K} \text{sg}(A_{b,k}) \sum_{j=1}^{L_{b,k}} \log p_\vartheta(\widehat{y}_{b,k,j} \mid x_b, \widehat{y}_{b,k,<j}). \tag{A.10}$$

The log-probabilities are recomputed at the functional student. Token log-probabilities are summed within a response and responses are averaged with denominator $BK$. All-correct and all-incorrect groups contribute zero advantages and remain in this denominator. Advantages are used without standardization. Responses, rewards, advantages, and response masks are held fixed in the derivative; the generation and verification operations supply the sampled data.

### A.4 Regularized teacher step and actual student update

The outcome meta-gradient is $h_t = \nabla_\phi F_t(\phi_t)$ from Equation (6). The teacher is regularized toward the initial privileged distribution using the unclipped KL

$$R_t(\phi) = \frac{1}{N_T} \sum_{(i,j)\in\mathcal{I}_T} D_{\text{KL}}\left(q_{\phi,ij}^{(T)} \,\middle\|\, q_{0,ij}^{(T)}\right), \qquad \widetilde{h}_t = h_t + \lambda \nabla_\phi R_t(\phi_t), \tag{A.11}$$

with $\lambda = 0.01$. Both distributions use the same privileged contexts, teaching prefixes, and distillation temperature. For one position, put $R_{ij} = \sum_v q_v \log(q_v/q_{0,v})$. Using the softmax Jacobian in Equation (A.5),

$$\begin{aligned}\frac{\partial R_{ij}}{\partial \ell_{\phi,u}} &= \frac{1}{T}\sum_v q_v(\mathbf{1}\{v=u\} - q_u)\left(\log\frac{q_v}{q_{0,v}} + 1\right) \\ &= \frac{q_u}{T}\left(\log\frac{q_u}{q_{0,u}} - R_{ij}\right).\end{aligned} \tag{A.12}$$

The constant terms cancel because $\sum_v q_v = 1$. At initialization $q_\phi = q_0$, so the anchor gradient is zero.

Let $U_t$ denote the teacher's independent optimizer state. One AdamW step, with the algebra of Equation (A.9) applied to $\phi_t, U_t, \widetilde{h}_t$, gives

$$(\phi_{t+1}, U_{t+1}) = \mathrm{AdamW}_{\eta_\phi}(\phi_t, U_t, \widetilde{h}_t). \tag{A.13}$$

The teacher uses $\eta_\phi = 10^{-5}$, betas $(0.9, 0.999)$, $\epsilon_a = 10^{-8}$, and zero weight decay. The outcome component $h_t$ and the anchor contribution are added before this optimizer step.

Finally, the updated teacher scores the same teaching prefixes and determines the actual student update:

$$(\theta_{t+1}, S_{t+1}) = \mathcal{U}(\theta_t, S_t, g_t(\phi_{t+1})) = (\theta'_t(\phi_{t+1}), S'_t(\phi_{t+1})). \tag{A.14}$$

Both evaluations of $\mathcal{U}$ start from the incoming state. The pilot supplies the teacher's learning signal; the state in Equation (A.14) is retained for the next cycle. Consequently the retained student step counter advances from $s$ to $s+1$ in each completed cycle, as in Algorithm 1.

Table A.1: MetaOPSD algorithm and training configuration.

| Component | Setting |
|---|---|
| Backbone | Qwen3-1.7B; Qwen3-4B for the main comparison |
| Student LoRA | Rank 64, alpha 128, dropout 0 |
| Student target projections | q, k, v, o, gate, up, down |
| Student AdamW | LR $5 \times 10^{-6}$; betas $(0.9, 0.999)$; $\epsilon = 10^{-8}$; weight decay 0 |
| Student gradient clipping / schedule | Global norm 0.1 / constant LR |
| Teacher residual | Rank 64; no bias; $A \sim \mathcal{N}(0, 0.02^2)$, $B = 0$ |
| Teacher AdamW | LR $10^{-5}$; same betas and epsilon; weight decay 0 |
| Teaching / feedback | 32 problems $\times$ 1 response / 4 problems $\times$ 4 responses |
| Teaching decoding | Cap 1024; temperature 1.1; top-$p$ 0.95; top-$k$ 20 |
| Feedback decoding | Cap 4096; temperature 1; top-$p$ 1; top-$k$ disabled |
| Prompt cap / mode | 2048 / non-thinking with step-by-step answer request |
| Distillation | Temperature 1.0; vocabulary-term upper clip 0.05; no $T^2$ factor |
| Feedback reduction | Sum response log-probabilities; mean all $BK$ trajectories |
| Meta horizon / anchor | $H = 1$ / $\lambda = 0.01$ |
| Teacher adaptation | One AdamW step using the regularized meta-gradient |
| Student updates / evaluation interval | 200 / every 40 updates; report step 200 |
| Training seeds | 42, 43, 44; three independent training runs |
| Result aggregation | Mean over the three independent training runs |
| Re-evaluation seed schedule | 42, 1042, 2042; 12 responses per problem per seed |
| Runtime | 8 GPUs; FSDP2; vLLM TP1; microbatch 1; gradient checkpointing |
| Precision | FP32 teacher residual / BF16 backbone and student |

# B Outcome Gradient and Optimizer-Mediated Differentiation

## B.1 Expected correctness after one student update

Fix the incoming student state, teaching responses, and feedback problems $\{(x_b, a_b)\}_{b=1}^B$. Write $f_t(\phi) = \theta'_t(\phi)$ for the parameter component of Equation (A.7). Define expected correctness on these

feedback problems by

$$V_F(\vartheta) = \frac{1}{B}\sum_{b=1}^{B} \mathbb{E}_{y\sim p_\vartheta(\cdot|x_b)}[r(y, a_b)]. \tag{B.1}$$

Here $p_\vartheta$ is the response distribution under the feedback generation protocol, including its stopping rule and length cap. The one-step correctness gain is $\Delta V_t(\phi) = V_F(f_t(\phi)) - V_F(\theta_t)$. Since the incoming student is fixed,

$$\nabla_\phi \Delta V_t(\phi) = \nabla_\phi V_F(f_t(\phi)). \tag{B.2}$$

Thus post-update correctness supplies the gradient of this conditional one-step teaching gain. The checkpoint probes in Section 4.3 measure learning gains through separate student continuations.

### B.2 RLOO ESTIMATES THE NEGATIVE OUTCOME GRADIENT

**Proposition 1** (Outcome gradient at the sampling student)**.** *Suppose the $K \geq 2$ responses for each feedback problem are sampled independently from $p_{f_t(\phi_t)}$, and all $K$ responses enter Equations* (3) *and* (4)*. Assume that response log-probabilities are scored under this same policy, differentiation can be exchanged with the response expectation, and $f_t$ is differentiable at $\phi_t$. Then, conditional on the fixed student state, teaching data, feedback problems, and current teacher,*

$$\mathbb{E}[h_t] = -\nabla_\phi V_F(f_t(\phi))|_{\phi=\phi_t} = -\nabla_\phi \Delta V_t(\phi_t). \tag{B.3}$$

*The expectation is over the sampled feedback responses.*

*Proof.* For a fixed problem $(x_b, a_b)$ and student $\vartheta$, the score-function identity is

$$\begin{aligned}\nabla_\vartheta \mathbb{E}_{y\sim p_\vartheta}[r(y, a_b)] &= \sum_y r(y, a_b)\nabla_\vartheta p_\vartheta(y \mid x_b) \\ &= \mathbb{E}_y[r(y, a_b)\nabla_\vartheta \log p_\vartheta(y \mid x_b)]\,.\end{aligned} \tag{B.4}$$

The derivative acts on response probabilities, so the verifier reward can be discrete. The trajectory score equals the sum of the scores of its generated tokens, which gives the token reduction in Equation (A.10).

At $\vartheta = f_t(\phi_t)$, denote the $k$th trajectory score by $\sigma_{b,k} = \nabla_\vartheta \log p_\vartheta(\widehat{y}_{b,k} \mid x_b)$ and its leave-one-out baseline by $\bar{r}_{b,-k} = (K-1)^{-1}\sum_{\ell\neq k} r_{b,\ell}$. The score has mean zero:

$$\mathbb{E}[\sigma_{b,k}] = \sum_y \nabla_\vartheta p_\vartheta(y \mid x_b) = \nabla_\vartheta 1 = 0. \tag{B.5}$$

Independence of the responses makes $\bar{r}_{b,-k}$ independent of $\sigma_{b,k}$, conditional on the problem and sampling student. Consequently,

$$\begin{aligned}\mathbb{E}[\bar{r}_{b,-k}\sigma_{b,k}] &= 0, \\ \mathbb{E}[A_{b,k}\sigma_{b,k}] &= \mathbb{E}[r_{b,k}\sigma_{b,k}] = \nabla_\vartheta \mathbb{E}_y[r(y, a_b)].\end{aligned} \tag{B.6}$$

Differentiating the fixed-response loss, averaging over the complete groups, and taking expectations therefore gives

$$\mathbb{E}\Big[\nabla_\vartheta J_F(\vartheta)|_{\vartheta=f_t(\phi_t)}\Big] = -\nabla_\vartheta V_F(\vartheta)|_{\vartheta=f_t(\phi_t)}\,. \tag{B.7}$$

The Jacobian $Df_t(\phi_t)$ depends on the fixed teaching data and incoming state, and is constant with respect to the feedback sampling. Applying the chain rule yields

$$\begin{aligned}\mathbb{E}[h_t] &= Df_t(\phi_t)^\top \mathbb{E}[\nabla_{\theta'} J_F(f_t(\phi_t))] \\ &= -Df_t(\phi_t)^\top \nabla_{\theta'} V_F(f_t(\phi_t)) = -\nabla_\phi V_F(f_t(\phi))|_{\phi=\phi_t}\,.\end{aligned} \tag{B.8}$$

Equation (B.2) gives the gain identity. □

For the group mean $\bar{r}_b = K^{-1}\sum_k r_{b,k}$, the advantages satisfy $A_{b,k} = K(r_{b,k} - \bar{r}_b)/(K-1)$ and $\sum_k A_{b,k} = 0$. The factor $K/(K-1)$ is the correction that removes the current response from its own baseline. Groups with constant rewards have zero contribution and are included in the expectation above.

Proposition 1 applies to the outcome component $h_t$ at the student that generated the feedback. The teacher optimizer subsequently acts on the regularized gradient $\widetilde{h}_t$ in Equation (A.11). The scalar $J_F$ is a fixed-response surrogate; correctness gains are measured using generated responses from the corresponding students. If feedback problems are additionally sampled from a distribution $\mathcal{D}$, taking their expectation replaces $V_F$ by $V_{\mathcal{D}}(\vartheta) = \mathbb{E}_{(x,a)\sim\mathcal{D}}\mathbb{E}_{y\sim p_\vartheta}[r(y,a)]$ under the same conditions.

### B.3 Factorization of the one-step meta-gradient

Separate the student update into its teaching gradient, global clipping, and AdamW parameter map:

$$\phi \longmapsto g_t(\phi) \longmapsto C(g_t(\phi)) \longmapsto f_t(\phi) \longmapsto J_F(f_t(\phi)). \tag{B.9}$$

At the pilot, define

$$G_t = \frac{\partial g_t}{\partial \phi}, \qquad C_g = \frac{\partial \bar{g}}{\partial g}, \qquad K_u = \frac{\partial \theta'}{\partial \bar{g}}, \qquad v_\theta = \nabla_{\theta'} J_F. \tag{B.10}$$

In particular, $G_t \in \mathbb{R}^{d_\theta \times d_\phi}$ is the mixed derivative of the teaching loss in Equation (A.6).

**Proposition 2** (One-step vector–Jacobian products)**.** *On a locally differentiable branch of the distillation and student update maps, with the incoming state and sampled data held fixed,*

$$h_t = G_t^\top C_g^\top K_u^\top v_\theta. \tag{B.11}$$

*It can be computed by successive vector–Jacobian products (VJPs) without constructing these Jacobian matrices.*

*Proof.* Applying the chain rule to Equation (B.9) gives $Df_t = K_u C_g G_t$. Transposing this product and multiplying by the outer gradient gives Equation (B.11). Writing the intermediate vectors as

$$v_u = K_u^\top v_\theta, \qquad v_g = C_g^\top v_u, \qquad h_t = \nabla_\phi \left\langle g_t(\phi), \mathrm{sg}(v_g) \right\rangle\big|_{\phi=\phi_t} \tag{B.12}$$

gives the VJP computation, since differentiation of the final scalar inner product with its second argument fixed equals $G_t^\top v_g$. □

The stop-gradient in Equation (B.12) fixes the vector passed to this last VJP at its evaluated value. All dependence of the optimizer map on the teaching gradient has already been included in $v_g$. The following derivatives make the two optimizer factors explicit.

### B.4 Derivative through global norm clipping

On the inactive branch of Equation (A.8), $C_g = I$. On the active branch with $n = \|g\|_2 > 0$,

$$\nabla_g \rho(g) = -\frac{\tau}{n(n+\epsilon_c)^2} g, \qquad C_g = \rho I - \frac{\tau}{n(n+\epsilon_c)^2} g g^\top. \tag{B.13}$$

This follows by differentiating $C(g) = \rho(g) g$ and using $\nabla_g n = g/n$. The resulting VJP is

$$C_g^\top v_u = \begin{cases} v_u, & n + \epsilon_c < \tau, \\ \rho v_u - \dfrac{\tau \langle g, v_u \rangle}{n(n+\epsilon_c)^2} g, & n + \epsilon_c > \tau. \end{cases} \tag{B.14}$$

The second term accounts for the dependence of the common clipping coefficient on every student parameter. With $\epsilon_c$ set to zero, the active-branch Jacobian becomes $\rho(I - gg^\top/n^2)$, projecting away the radial component. The actual calculation retains $\epsilon_c$ as above.

### B.5 Derivative through the current AdamW step

For Equation (A.9), set $a_1 = (1-\beta_1)/b_1$, $a_2 = (1-\beta_2)/b_2$, and $d_i = \sqrt{\widehat{v}_i} + \epsilon_a$ for coordinate $i$. When $\widehat{v}_i > 0$,

$$\frac{\partial \widehat{m}_i}{\partial u_i} = a_1, \qquad \frac{\partial \widehat{v}_i}{\partial u_i} = 2 a_2 u_i, \qquad \frac{\partial d_i}{\partial u_i} = \frac{a_2 u_i}{\sqrt{\widehat{v}_i}}. \tag{B.15}$$

The quotient rule gives

$$k_i := \frac{\partial \theta_i'}{\partial u_i} = -\eta_\theta \left[ \frac{a_1}{d_i} - \frac{\widehat{m}_i a_2 u_i}{\sqrt{\widehat{v}_i}\, d_i^2} \right], \qquad K_u^\top v_\theta = k \odot v_\theta. \tag{B.16}$$

The weight-decay term has zero derivative with respect to $u$ because $\theta_t$ is fixed. The second term in brackets differentiates the current second-moment denominator. Historical moments affect both $\widehat{m}$ and $\widehat{v}$, while their incoming values remain fixed inputs. AdamW acts coordinatewise on $u$; the global coupling has been accounted for by Equation (B.14).

For an initially unused coordinate with $m_i = v_i = u_i = 0$, the composite parameter update has derivative $-\eta_\theta a_1/\epsilon_a$ at zero. Indeed its update term is $-\eta_\theta a_1 u_i/(\sqrt{a_2}|u_i| + \epsilon_a)$, whose difference quotient converges to that value. The positive-variance expression in Equation (B.16) is evaluated only for $\widehat{v}_i > 0$.

### B.6 Evaluation of the mixed derivative and differentiation scope

The last VJP uses the teacher dependence derived in Equation (A.5). For microbatch or device partition $r$ with teaching positions $\mathcal{I}_T^{(r)}$, define

$$\begin{aligned} g_t^{(r)}(\phi) &= \frac{1}{N_T} \sum_{(i,j)\in\mathcal{I}_T^{(r)}} \nabla_\theta L_{ij}(\theta,\phi)|_{\theta=\theta_t}, \qquad g_t(\phi) = \sum_r g_t^{(r)}(\phi), \\ h_t &= \sum_r \nabla_\phi \langle g_t^{(r)}(\phi), \mathrm{sg}(v_g)\rangle \Big|_{\phi=\phi_t}. \end{aligned} \tag{B.17}$$

Each contribution uses the global token denominator $N_T$ and the same $v_g$ computed from the aggregated teaching gradient. This decomposition allows the teaching graph to be reconstructed one microbatch at a time. The feedback gradient analogously uses the global trajectory denominator $BK$ in Equation (A.10).

These derivatives unroll one student update. The incoming parameters, optimizer history, frozen backbone, sampled tokens, and verifier outputs are fixed, while the teacher residual, teaching gradient, clipping coefficient, and current AdamW moments remain differentiable. The teacher optimizer step is applied after $h_t$ has been computed. At vocabulary clipping thresholds $d_v = c$ and the global clipping boundary $n + \epsilon_c = \tau$, derivatives follow the implementation's branch convention; the propositions apply on differentiable branches. The formulas describe the real-arithmetic update, with FP32 teacher and loss calculations and BF16 backbone and student execution as listed in Table A.1.

## C Data, Prompts, and Evaluation

### C.1 Data roles and benchmark snapshots

Training uses the mathematical reasoning subset of OpenThoughts with the problem–solution budget specified in Section 4.1. Teaching and feedback batches are disjoint by problem within each cycle; both are drawn from the training pool. The teacher receives the reference solution, while the student generates from the problem alone. Verifier targets are used for grading and excluded from generation prompts.

The three final benchmarks contain 90 problems in total. The teaching-utility study uses a fixed set of 128 MATH-500 questions, excluded from training, feedback, development, and the final benchmarks, as specified in Section 4.3. Table C.1 identifies the dataset snapshots. The MATH-500 entry identifies the source collection for the probe.

### C.2 Prompt construction and response alignment

The following training-message templates substitute the example's problem and reference solution before applying the model's chat template.

Table C.1: Dataset snapshots pinned by the evaluation manifest. Both model sizes use the same three benchmarks. The MATH-500 row identifies the source collection for the 128-question teaching-utility probe.

| Dataset, source repository, and split | Pinned revision |
|---|---|
| AIME24: 30 problems; `train` `HuggingFaceH4/aime_2024` | `2fe88a2f1091d5048c0f36abc874fb997b3dd99a` |
| AIME25: 30 problems; `train` `yentinglin/aime_2025` | `6f71d77b0b89b9dabe07ab466c51df33f514df7f` |
| HMMT25: 30 problems; `train` `MathArena/hmmt_feb_2025` | `6fdc4277120810ff75aa22d2d5489b91f7a262a1` |
| MATH-500: 500 source problems; `test` `HuggingFaceH4/MATH-500` | `6e4ed1a2a79af7d8630a6b768ec859cb5af4d3be` |

**Student and outcome-feedback user message.**

```
Problem: [problem]

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

**Privileged-teacher user message.**

```
Problem: [problem]

Here is a reference solution to this problem:
=== Reference Solution Begin ===
[reference solution]
=== Reference Solution End ===

After reading the reference solution above, make sure you
truly understand the reasoning behind each step -- do not
copy or paraphrase it. Now, using your own words and
independent reasoning, derive the same final answer to the
problem above. Think step by step, explore different
approaches, and don't be afraid to backtrack or reconsider
if something doesn't work out:

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

During distillation, the collected student response is appended as the assistant continuation to both prompts. The teacher and student can have different prompt lengths; their next-token distributions are aligned by response position $j$. Prompt and padding positions are masked out. Reference-NLL feedback scores the reference continuation under a problem-only user message.

Training uses non-thinking mode with an explicit reasoning request. Benchmark evaluation uses the user message

```
[problem]

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

This message is passed as the user role to the tokenizer's chat template with a generation prompt and thinking mode enabled.

### C.3 Decoding, answer verification, and repeat seeds

Each evaluated checkpoint generates 12 responses per benchmark problem. Evaluation uses BF16 precision, thinking mode, temperature 1, top-$p$ 1, disabled top-$k$ filtering, and min-$p$ 0. Evaluation checkpoints are student updates $40, 80, 120, 160, 200$; benchmark tables report update 200.

**Verifiable answer scoring.** The evaluator locates the final `\boxed` occurrence and requires a complete, nonempty braced answer. Missing, empty, or incomplete final boxes receive zero under the strict metric. The extracted answer is compared with the verifier target using `math_verify`. If this does not establish a match, the evaluator also checks normalized exact equality after removing formatting commands, whitespace, and trailing punctuation and applying case folding. A response that reaches the token budget can still receive credit for a valid correct final box.

**Fixed seed schedule for re-evaluation.** For reproducible re-evaluation, we specify parent generation seeds 42, 1042, and 2042, with offsets $0, \ldots, 11$ for the 12 responses. These control response sampling from each checkpoint. The reported three-seed aggregation averages independent training runs with seeds 42, 43, and 44 (Table A.1).

### C.4 Score aggregation

Let $\mathcal{S}_{\text{train}} = \{42, 43, 44\}$ index the independent training runs, and let $r_{s,e,d,b,j} \in \{0, 1\}$ denote correctness for response $j$ to problem $b$ of benchmark $d$ in generation repeat $e$ of training run $s$. With $N_d$ problems and $E_s$ generation repeats for that trained checkpoint, first form its evaluation score, then average across training runs:

$$a_{s,d} = \frac{100}{12 N_d E_s} \sum_{e=1}^{E_s} \sum_{b=1}^{N_d} \sum_{j=1}^{12} r_{s,e,d,b,j},$$
$$\bar{a}_d = \frac{1}{3} \sum_{s \in \mathcal{S}_{\text{train}}} a_{s,d}. \tag{C.1}$$

Thus avg@12 averages individual response correctness within each run, and the three independently trained models receive equal weight. The re-evaluation schedule above uses $E_s = 3$ for each checkpoint. Macro accuracy is the unweighted mean of AIME24, AIME25, and HMMT25. Method differences are computed in percentage points, and aggregation precedes rounding. Teacher contrasts are formed within the same student state before averaging across the five student checkpoints, as specified in Appendix D.1.

## D Probe and Comparison Protocols

### D.1 Checkpoint cross-evaluation summaries

The diagnostic crosses student checkpoints $\mathcal{S} = \{40, 80, 120, 160, 200\}$ with teacher checkpoints $\mathcal{T} = \{0, 40, 80, 120, 160, 200\}$, yielding 30 student–teacher combinations. At each student checkpoint $s$, all six branches restore the same $\theta_s$, AdamW moments, and step counters. Each branch fixes its assigned teacher for $H = 20$ OPSD updates, using the student optimizer and constant learning rate $5 \times 10^{-6}$ in Table A.1. These diagnostic continuations run on copies of the student and leave the main training endpoint unchanged.

The branches share the ordered teaching problems and generate their own on-policy responses as their student states evolve. The unchanged student and all six endpoints are evaluated on the same 128 MATH-500 questions with 12 responses per question and the three-training-seed averaging in Section 4.3. This diagnostic horizon $H$ specifies additional student learning; the differentiated pilot used during MetaOPSD training has horizon one.

Figure 5 reports individual utilities $U_{s,t}$ and the equal-weight mean of $U_{s,t} - U_{s,0}$ over the five student states, including teacher checkpoints later than the student checkpoint. Row maxima include ties. These checkpoint summaries are descriptive; dispersion across checkpoint pairs is not an uncertainty interval.

### D.2 Teacher-update frequency

The frequency study in Figure 4 uses $k \in \{1, 5, 10, 20\}$ with 200 actual student updates per run. At student steps divisible by $k$, apply Algorithm 1; otherwise, take one OPSD step under the current teacher. Teacher parameters and optimizer state stay fixed between adaptations. The four intervals

give 200, 40, 20, and 10 teacher updates, respectively, with 16 feedback responses and a one-step pilot per update.

The branches share initialization, optimizer settings, teaching-problem order, and per-cycle feedback budget. Feedback problems are indexed by absolute student step. Teaching and feedback use separate random streams; each branch generates responses with its own current student. Evaluation follows the 40-step schedule in Table A.1. Total feedback and computation vary with the adaptation interval.

### D.3 Reference-NLL scoring and learning curves

**Reference scoring and differentiation.** Prompt and padding tokens are excluded from the loss in Equation (10). Reference scoring uses temperature one, with the same serialization and terminal-token convention across runs. Length filtering is fixed for the shared feedback pool before training. The teacher minimizes $F_t^{\mathrm{ref}}(\phi) = J_{\mathrm{ref}}(\theta'_t(\phi))$ with meta-gradient

$$h_t^{\mathrm{ref}} = \left(\frac{\partial \theta'_t}{\partial \phi}\right)^{\top} \nabla_{\theta'} J_{\mathrm{ref}}(\theta'), \tag{D.1}$$

evaluated at $\phi = \phi_t$ and $\theta' = \theta'_t(\phi_t)$. We retain differentiation through the pilot optimizer step, holding the incoming student parameters, optimizer history, teaching responses, and reference targets fixed. Teacher parameterization, regularization, and optimizer settings follow the main method. Reference scoring supplies feedback without sampling responses or forming RLOO advantage groups.

**Experimental protocol.** We reuse the Qwen3-1.7B OPSD and MetaOPSD runs from Section 4.2. The reference-NLL run follows the matched 200-update protocol in Section 4.4, including the shared teacher settings and teaching- and feedback-problem streams. Each branch generates its own on-policy teaching responses. The four feedback problems per cycle come from the training pool and are disjoint from the teaching batch; the probe and final benchmarks are excluded.

Figure D.1 reports Macro accuracy every 40 updates using the evaluation settings in Section 4.1. Its step-200 scores match Table 2. Checkpoints are repeated measurements within a training run.

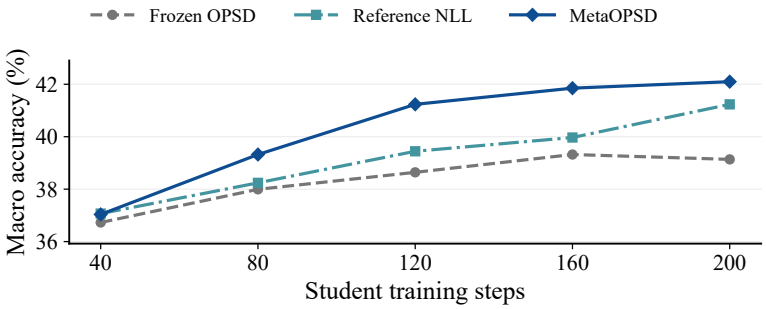


Figure D.1: **Reference-NLL learning curves.** Macro accuracy every 40 student updates for the reference-NLL variant, OPSD, and MetaOPSD, computed as the unweighted mean of AIME24, AIME25, and HMMT25 at each checkpoint. All runs use 200 student updates. Only supplied means are shown.

### D.4 Baseline training settings

SFT uses AdamW with learning rate $10^{-5}$ and effective batch size 32, fitting the OpenThoughts chain-of-thought solutions from problem-only prompts. GRPO uses 16 problems per rollout batch and eight responses per problem (128 responses per batch), with a 4,096-token response limit and the same binary correctness verifier as MetaOPSD. Frozen OPSD and MetaOPSD share the clipped distillation objective, with temperature 1.0 and vocabulary-term upper clip 0.05. Student and teacher optimizer settings for MetaOPSD are listed in Table A.1.