THE MISSING PRIMITIVE: DIAGNOSING AND REPAIRING MATHEMATICAL REASONING IN LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

While Large Language Models (LLMs) have demonstrated striking capabilities on frontier mathematical problems, it remains unclear whether they possess the structural mathematical understanding underlying their solutions. In this paper, we take a first step toward systematically studying mathematical understanding in LLMs, from diagnosing its distinct capabilities to leveraging these findings to improve post-training. First, we introduce the notion of Mathematical Primitive to probe structural mathematical understanding and propose PRIM, a novel benchmark that evaluates mathematical reasoning along four distinct dimensions: Discovery, Generation, Digestion, and Execution. Second, our systematic diagnosis shows that solution accuracy masks distinct capability profiles, primitives unlock substantial latent execution capacity, and Discovery is the dominant bottleneck in mathematical reasoning. Our post-training analysis further shows that discovery-limited failures are particularly amenable to repair. Finally, building on these findings, we introduce ABSORB, a primitive-privileged self-distillation framework that selectively transfers primitive-guided reasoning into the student model. Extensive experiments demonstrate that ABSORB consistently improves mathematical reasoning over baselines across model scales and challenging benchmarks.

1 INTRODUCTION

Recent advances in Large Language Models (LLMs) have driven rapid progress in mathematical problem solving, with frontier models increasingly capable of tackling research-level questions and, in several recent cases, resolving or making substantive progress on long-standing open problems (OpenAI, 2026a;b; Claude, 2026; Bubeck et al., 2025; Jang & Ryu, 2025). Yet these striking successes leave unresolved a more foundational question: what models actually understand about the problems they solve and the mathematical content of the solutions they produce. Successful problem solving does not necessarily imply such understanding: models may arrive at correct answers through specious deductions (Boye & Moell, 2025), while conversely identifying sound high-level proof structures yet failing in low-level technical execution (Lu et al., 2024; Qin et al., 2025). Existing works provide limited insight into this disconnect, as mathematical reasoning is still assessed primarily through outcome-level metrics such as final-answer accuracy or formal proof verifiability, obscuring whether success reflects the discovery of the mathematical structure needed for a solution or merely the successful execution of a derivation.

Understanding mathematical reasoning therefore requires more than verifying that a derivation is correct; it requires exposing the ideas that organize the argument—why it works, which components are essential, and what can be generalized or reused. This need becomes increasingly salient as LLM-generated proofs grow in scale and complexity: lengthy chains of valid steps may establish correctness while leaving the underlying mathematical structure difficult to identify. Recent work has begun to emphasize precisely these aspects of mathematical reasoning, calling for representations that reveal the motivation, structure, and key ideas underlying an argument rather than only its final outcome (Frieder et al., 2024). In his ICM 2026 lecture, Terence Tao similarly highlighted the importance of making machine-generated mathematics conceptually digestible (Tao, 2026), warning that optimizing primarily for formal correctness risks producing an increasing volume of valid but opaque proofs without a corresponding increase in mathematical understanding. For LLMs, this

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.

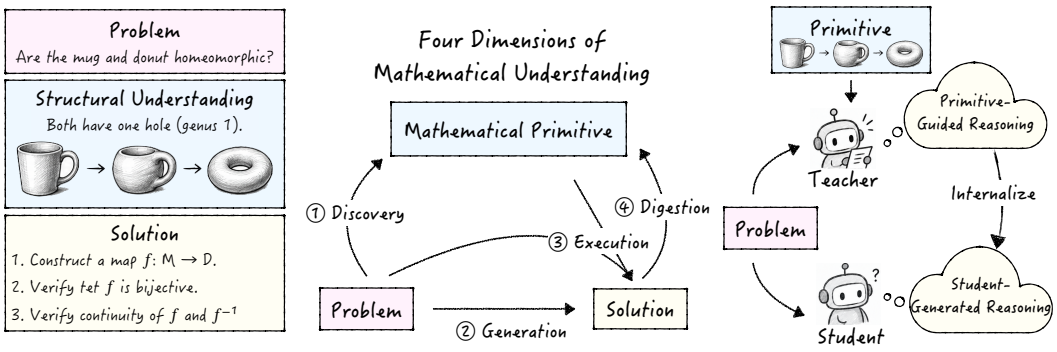


Figure 1: **Diagnosing and internalizing mathematical primitives.** We introduce Mathematical Primitives to probe structural mathematical understanding in LLMs across four dimensions: Discovery, Generation, Digestion, and Execution. Our diagnosis further motivates ABSORB, which selectively transfers primitive-guided reasoning into the student without requiring primitives at inference time.

perspective shifts the focus from merely producing valid proofs to determining whether the structural knowledge organizing those proofs is actually accessible to the model.

In this paper, we systematically study structural mathematical understanding in LLMs, as illustrated in Figure 1: the ability to identify the compact, load-bearing mathematical structure that explains *why a solution works*, distinct from the procedural ability to generate the derivation itself. We operationalize this notion through the **Mathematical Primitive**, a concise, non-procedural representation of the foundational idea organizing a solution, and introduce PRIM, a benchmark spanning four complementary dimensions: Discovery, Generation, Digestion, and Execution. Our diagnosis reveals that ❶ **similar solving accuracy can mask markedly different capability profiles**, ❷ **correct primitives unlock substantial latent execution capacity**, and, most importantly, ❸ **independent Discovery is the dominant bottleneck** to mathematical reasoning. We further find that discovery-limited failures are substantially more amenable to post-training, but existing methods can introduce regressions on problems the model already solves. Motivated by these findings, we introduce ABSORB, a post-training paradigm that uses Mathematical Primitives as privileged information and a bounded override mechanism to selectively transfer their guidance into the student's reasoning process, enabling primitive-guided reasoning to be internalized without access to primitives at inference time. Our main contributions are as follows:

- **Benchmarking structural mathematical understanding.** We introduce the notion of Mathematical Primitive and propose PRIM, a benchmark that evaluates mathematical reasoning across four complementary dimensions: Discovery, Generation, Digestion, and Execution.
- **Systematic Diagnosis.** Our analysis reveals distinct capability profiles hidden by answer accuracy, substantial latent execution capacity unlocked by correct primitives, and independent Discovery as the dominant bottleneck in mathematical reasoning. We further show that discovery-limited failures are substantially more amenable to post-training.
- **Internalizing primitive-guided reasoning.** Building on these findings, we introduce ABSORB, a primitive-privileged post-training paradigm that selectively transfers primitive-guided reasoning into the student model. Across multiple model scales and challenging mathematical reasoning benchmarks, ABSORB consistently improves average performance by 2.42 – 4.48 points.

2 MATHEMATICAL PRIMITIVES AS STRUCTURAL UNDERSTANDING

Rather than viewing mathematical problem solving as a sequence of steps, we separate **structural mathematical understanding** from **procedural execution**. We formalize the distinction through the **Mathematical Primitive** (§2.1) and introduce PRIM, a benchmark that evaluates LLM mathematical reasoning across four dimensions: Generation, Discovery, Digestion, and Execution (§2.3).

2.1 MATHEMATICAL PRIMITIVE

A mathematical solution often begins before the first line of a proof, when an opaque problem is recognized through the structure that makes it solvable. We term this load-bearing structural understanding the **Mathematical Primitive** (Definition 1). Unlike generic advice such as use

"induction" or "apply a theorem", a primitive identifies the structural property that makes such a strategy applicable. It therefore serves as an intermediate reasoning object between problem understanding and procedural solution construction.

Definition 1 (Mathematical Primitive)**.** *For a mathematical problem, a* **primitive** *is the essential conceptual observation that reveals* **why** *the problem can be solved. It captures the latent mathematical structure shared by the problem and its solution, such as an invariant, a theorem condition, a representation, a reduction, or a reformulation. A primitive should be* **concise**, **problem-specific**, *and* **explanatory***: it must make clear what property is being exploited and how that property unlocks the solution. It is* **not** *a routine calculation, a piece of generic advice, or a full proof.*

2.2 Four Dimensions of Mathematical Understanding

Taking the mathematical primitive as the fundamental concept of structural understanding, we systematically decompose mathematical reasoning into four complementary dimensions. Let x denote the input problem, p the corresponding mathematical primitive, y a complete proof or solution, and π an LLM solver. The four dimensions are defined as follows:

- **Discovery** ($\pi(x) \to \hat{p}$) measures whether the model can independently identify the load-bearing mathematical primitive from the input problem alone.
- **Generation** ($\pi(x) \to \hat{y}$) evaluates whether the model can solve the problem directly under zero-shot CoT prompting, with performance measured by exact match on the final answer.
- **Digestion** ($\pi(x, y) \to \hat{p}$) instead provides the model with a complete correct solution and asks it to extract the mathematical primitive underlying the derivation, measuring whether the model can identify the underlying structure once a valid reasoning trajectory is exposed.
- **Execution** ($\pi(x, p) \to \hat{y}$) conditions the solver on the mathematical primitive and evaluates whether it can successfully solve the problem once the relevant structure is explicitly provided.

Both Discovery and Digestion are evaluated using the scoring protocol introduced in § 2.3.

2.3 The PRIM Benchmark

Data Curation. We build on Humanity's Last Exam (HLE) (Center for AI Safety et al., 2026), focusing on its mathematics, text-only, free-form subset with `exactMatch` evaluation. To reduce annotation noise in the original benchmark, we draw from the HLE-Verified release (Zhai et al., 2026), retaining its *Gold* and *Revision* subsets, from which we randomly sample 200 problems. For each problem, GPT-5.4-High generates an initial primitive annotation conditioned on the problem, reference answer, and gold rationale, following Definition 1. Three human experts with graduate-level training in mathematics independently review the primitive and verify the validity of the problem, answer, and rationale. This process excludes 18 unsuitable problems, yielding a final evaluation set of 182. Further curation and annotation details are provided in Appendix A.1.

Primitive Scoring. We evaluate each predicted primitive against the human-annotated gold primitive, using the reference solution only to recognize mathematically equivalent formulations. To account for different forms of mathematical structure, each problem is assigned to one of three primitive families, including Recast, Witness, and Argument, with full definitions provided in Appendix A.2.

Our scoring rubric evaluates three aspects of a predicted primitive: V determines whether the response constitutes a valid primitive, σ_{gate} measures whether it identifies the essential mathematical structure, and σ_{mech} measures whether it explains how that structure enables the solution. We define

$$\text{Score} = V \cdot \sigma_{\text{gate}} \cdot \big(0.6 + 0.4 \cdot \sigma_{\text{mech}}\big), \quad \text{PrimitiveAcc} = \mathbf{1}\left[\text{Score} \geq \tau\right],$$

with $\tau = 0.8$. This formulation treats validity and structural identification as necessary conditions, while using mechanism quality to refine the score. Further details are provided in Appendix A.3.

3 Diagnosing Mathematical Reasoning

In this section, we conduct an empirical study of a diverse set of LLMs, including both open- and closed-source models, on PRIM. Moving beyond standard answer correctness, our evaluation systematically disentangles procedural execution from structural understanding. We report model

performance across four dimensions: **Generation**, **Discovery**, **Digestion**, and **Execution**, following the evaluation protocol described in § 2.3. The main results are summarized in Table 1.

3.1 Evaluation Setup

We evaluate 12 models spanning three lineages and a broad capability range. The OpenAI lineage includes the closed-source GPT-5.4 (OpenAI, 2026a), GPT-5.4-Mini (OpenAI, 2026b), and GPT-5.4-Nano (OpenAI, 2026b), alongside the open-weight GPT-OSS-20B (OpenAI, 2025). The Qwen lineage comprises Qwen-3.6-27B (Qwen Team, 2026b) and Qwen-3.5-{27B, 9B, 4B} (Qwen Team, 2026a). The DeepSeek lineage features DeepSeek-R1-0528-Qwen3-8B (DeepSeek-AI, 2025) and DeepSeek-R1-Distill-Qwen-{32B, 14B, 7B} (DeepSeek-AI, 2025). We use high reasoning effort when available and a maximum generation budget of 120k tokens per problem.

Model	**Discovery**	**Generation**	**Digestion** Dig. − Disc.	**Execution** Exec. − Gen.
OpenAI				
gpt-5.4	82.42	67.03	100.00 (+17.58)	86.81 (+19.78)
gpt-5.4-mini	59.34	50.55	97.80 (+38.46)	71.98 (+21.43)
gpt-5.4-nano	41.76	44.51	97.25 (+55.49)	68.68 (+24.17)
gpt-oss-20B	34.62	43.41	91.21 (+56.59)	60.99 (+17.58)
Qwen				
Qwen3.6-27B	24.73	52.75	92.31 (+67.58)	78.57 (+25.82)
Qwen3.5-27B	28.57	47.80	93.41 (+64.84)	71.98 (+24.18)
Qwen3.5-9B	13.19	38.46	79.67 (+66.48)	61.54 (+23.08)
Qwen3.5-4B	6.59	26.37	70.88 (+64.29)	56.04 (+29.67)
DeepSeek-R1 Distill				
R1-0528-8B	6.59	17.03	68.68 (+62.09)	39.56 (+22.53)
R1-Distill-32B	6.04	18.13	65.38 (+59.34)	43.41 (+25.28)
R1-Distill-14B	4.40	15.93	67.03 (+62.63)	39.01 (+23.08)
R1-Distill-7B	7.14	13.19	48.90 (+41.76)	32.42 (+19.23)

Table 1: Overall performance on the four dimensions of PRIM. Detailed results are presented in Table 7 in Appendix B.3.

3.2 Results Analysis

Table 1 presents the comprehensive evaluation results of all models on PRIM. Performance in the Discovery and Digestion dimensions is assessed by the model's capacity to isolate the underlying mathematical primitive, utilizing the scoring protocol detailed in § 2.3. Meanwhile, Generation and Execution are evaluated based on standard final-answer accuracy, following the HLE benchmark (Center for AI Safety et al., 2026). Further details on the diagnostic evaluation and additional analyses are provided in Appendix B.

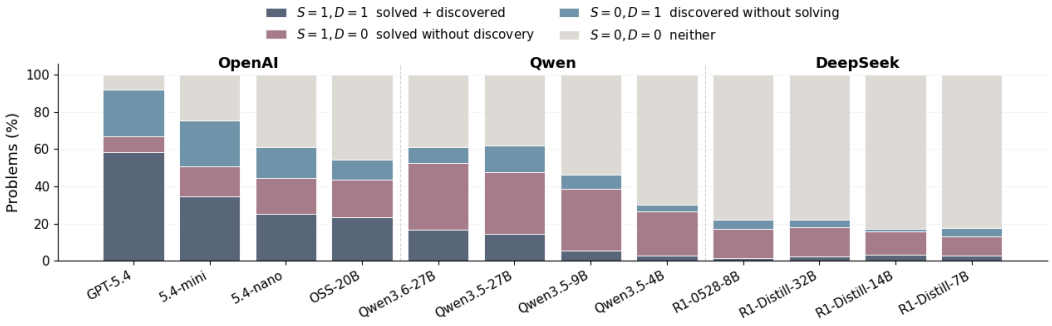


Figure 2: Joint composition of Generation and Discovery outcomes across models.

Finding 1: Answer accuracy masks distinct capability profiles. Final-answer accuracy conflates structural discovery with downstream execution. As shown in Figure 2, Discovery and Generation are positively associated overall, yet models with similar solving accuracy can exhibit sharply different structural capabilities. For example, Qwen3.6-27B and gpt-5.4-mini achieve comparable Generation performance, while differing by over 30 percentage points in Discovery. Conversely, GPT-5.4 achieves substantially higher Discovery than Generation. The problem-level decomposition in Figure 2 reveals similarly distinct profiles across model families. The OpenAI lineage is more **primitive-forward**, with successful Generation frequently accompanied by correct Discovery and a substantial fraction of cases with correct Discovery but failed Generation (D^+G^-). In contrast, Qwen

exhibits more cases with failed Discovery but successful Generation (D^-G^+), despite competitive Generation performance. The DeepSeek-R1 distilled models are more heavily concentrated in joint Discovery–Generation failures. Thus, similar top-line accuracy can arise from markedly different underlying capability profiles.

Qualitative analysis further clarifies these asymmetries. For gpt-5.4, D^+G^- cases primarily reflect two post-primitive failure modes: ❶ **technical knowledge gaps**, where the correct structure is identified but a required lemma, theorem condition, or domain-specific fact is missing; and ❷ **procedural execution failures**, such as algebraic errors, missed edge cases, or unresolved logical gaps. For Qwen3.6-27B, among D^-G^+ cases, the decisive structural idea often emerges after an initial attempt without using it, typically around one quarter into the reasoning trace; in most remaining cases it appears at the outset, and only rarely is it absent entirely. The corresponding cold Discovery output often captures a partial form of the correct idea but misses its decisive component. Together with the longer reasoning traces in Figure 12, this pattern is consistent with a more **grind-first** profile, where the relevant structure tends to **emerge during extended procedural exploration** rather than being isolated before execution.

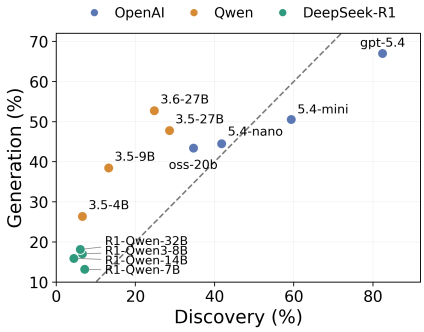


Figure 3: Discovery versus Generation across all evaluated models on PRIM.

Finding 2: Correct primitives unlock latent execution capacity. To examine whether models possess downstream capabilities that are not realized during direct Generation, we evaluate Execution when the correct Mathematical Primitive is explicitly provided. As shown in Table 1, primitive guidance improves performance by 17.58 to 29.67 percentage points across all 12 models. Notably, under the same primitive-guided setting, the Qwen models achieve Execution performance competitive with the closed-source gpt-5.4 variants despite substantially weaker Discovery. This gap shows that difficulty independently identifying the relevant mathematical structure can coexist with strong downstream problem-solving capability once that structure is available. More broadly, these gains indicate that many Generation failures mask downstream solving capacity that models can successfully exercise when the relevant mathematical structure is supplied.

Model	Generation	Execution			
		Self-Generated Primitive	Teacher Plan	Teacher Primitive	Gold Primitive
gpt-5.4-mini	50.55	48.90 (-1.65)	60.44 (+9.89)	63.74 (+13.19)	71.98 (+21.43)
gpt-5.4-nano	44.51	46.15 (+1.64)	54.40 (+9.89)	60.99 (+16.48)	68.68 (+24.17)
Qwen3.6-27B	52.75	41.21 (-11.54)	56.04 (+3.29)	64.84 (+12.09)	78.57 (+25.82)

Table 2: Generation and Execution performance on PRIM under different forms of reasoning support. Detailed results are presented in Table 8 in Appendix B.3.

We further investigate whether this rescue can be attributed simply to task decomposition or generic procedural scaffolding. Table 2 compares four intervention settings. Conditioning the model on its own generated primitive provides little benefit: it yields only a marginal gain for gpt-5.4-nano and substantially degrades Qwen3.6-27B, indicating that a forced discover-then-execute decomposition alone is insufficient. We then compare two forms of external guidance generated by the same teacher model (gpt-5.4) from the problem alone: a step-by-step **Teacher Plan**, which serves as a procedural hint by specifying the major solution steps, and a **Teacher Primitive**, which is generated according to the definition of primitive. Although the Teacher Plan improves over direct Generation, the Teacher Primitive yields substantially larger gains across the evaluated models. The Gold Primitive further maximizes this effect, pushing accuracy to 71.98%, 68.68%, and 78.57% for the representative models. This contrast provides crucial empirical validation for our premise in §2.1: a mathematical primitive is fundamentally distinct from, and vastly more effective than, a generic plan or hint. The rescue effect therefore cannot be attributed to task decomposition or the procedural scaffold evaluated here alone; exposing the load-bearing mathematical structure provides a substantially stronger signal for downstream reasoning.

Finding 3: Independent structural discovery is the dominant bottleneck. As shown in Table 1, models perform substantially better on **Digestion**, recovering the Mathematical Primitive

from a correct solution, than on **Discovery**, where the same primitive must be identified from the problem alone. This asymmetry is particularly pronounced outside the strongest OpenAI models. Even models with weak Discovery can often recover the organizing primitive once the relevant reasoning is exposed through a completed solution; for instance, Qwen3.6-27B rises from 24.73% Discovery to 92.31% Digestion and R1-Distill-32B from 6.04% to 65.38%. These results reveal a pronounced gap between retrospective recognition and prospective discovery of mathematical structure: models can often recognize the key idea once it is instantiated in a valid solution, yet struggle to identify it independently before the derivation is known.

To further localize this bottleneck, we decompose Generation failures according to the joint outcomes of Discovery (D) and Execution (E), as shown in Figure 4. A striking 83.6% of all failures fall into the D^- regime, indicating that failed primitive discovery dominates the failure landscape. Importantly, a substantial fraction of these cases are **discovery-limited** failures (D^-E^+): the model fails to identify the correct primitive independently, yet succeeds once it is supplied. This reveals that, for many failures, latent execution capacity is already present but remains inaccessible because of the upstream bottleneck in primitive discovery.

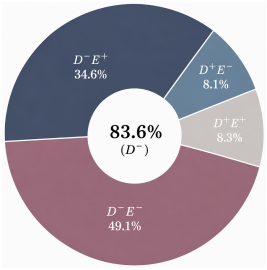


Figure 4: Decomposition of Generation failures by Discovery (D) and Execution (E). Detailed results can be found in Table 9 in Appendix B.3.

3.3 Further Analysis: Which Failures Are Repairable?

While our preceding diagnosis reveals the failure modes of LLMs during inference, a natural follow-up question arises: **how does post-training impact these mathematical reasoning failures?** Of particular interest are **discovery-limited** failures, which exhibit substantial room for improvement (§3.2). We hypothesize that these failures lie just at the boundary of the models' reasoning capabilities, making them highly amenable to repair through post-training. Therefore, we utilize Supervised Fine-Tuning (SFT) and On-Policy Self-Distillation (OPSD) (Zhao et al., 2026) to investigate how different failure modes uniquely respond to these post-training paradigms.

Setup. Our training corpus contains 709 Mathematics Ph.D. qualifying-examination problems from 1991–2026, each paired with a human-authored proof (median length: 118 words); additional details are provided in Appendix C. We use qualifying-examination problems because their open-ended, proof-oriented nature closely matches the difficulty and structural reasoning demands of PRIM. Since such problems generally lack algorithmically verifiable short answers, we focus on text-supervised post-training rather than outcome-verified methods such as RLVR. We evaluate SFT and OPSD on Qwen-3.5-27B, 9B, 4B (Qwen Team, 2026a), using the default OPSD configuration, bfloat16 precision, a fixed seed of 42, and NVIDIA RTX 6000 Ada GPUs.

On-Policy Self-Distillation. Given training examples $\{x, y\}$, where x denotes the input problem and y its reference solution, both the teacher and student policies are initialized from the same base model π_0. At each decoding step t, the student autoregressively generates an on-policy response, inducing the token distribution $P_t := \pi_S(\cdot \mid \hat{y}_{<t}, x)$, while the teacher distribution $Q_t := \pi_T(\cdot \mid \hat{y}_{<t}, x, y)$ additionally conditions on the reference solution. Notably, both policies are conditioned on the same student prefix $\hat{y}_{<t}$, allowing the teacher to provide token-level supervision along the student's own reasoning trajectory. OPSD minimizes the divergence between these distributions:

$$\mathcal{L}_{\text{OPSD}} = \frac{1}{T}\sum_{t=1}^{T} D(P_t, Q_t) = \frac{1}{T}\sum_{t=1}^{T} D\left(\pi_S(\cdot \mid \hat{y}_{<t}, x), \pi_T(\cdot \mid \hat{y}_{<t}, x, y)\right),$$

where D is instantiated as Kullback–Leibler (KL), reverse KL (rKL), or Jensen–Shannon (JSD). Specifically, KL uses $D_{\text{KL}}(Q_t \parallel P_t)$, rKL uses $D_{\text{KL}}(P_t \parallel Q_t)$, and JSD is symmetric in P_t and Q_t.

Results. Across the 27B, 9B, and 4B models, the base models leave 341 failure instances in PRIM. We focus on the 294 cases with failed Discovery: 147 D^-E^+ and 147 D^-E^- cases. We define repair as an incorrect baseline Generation response becoming correct after post-training. As shown in Table 3, SFT and OPSD repair 20.4% and 21.1% of D^-E^+ cases, roughly three times their respective rates of 6.8% and 8.2% for D^-E^- cases. This ordering holds at each model scale, identifying discovery-limited

Failure type	SFT	OPSD
D^-E^+	20.4%	21.1%
D^-E^-	6.8%	8.2%

Table 3: Post-training repair rates of SFT and OPSD across initial PRIM failure quadrants.

failures as a promising target for post-training. However, net gains in Generation remain modest (as shown in Table 4) because some initially correct cases become incorrect, offsetting part of these repairs. These regressions are consistent with post-training drift and suggest that supervision from reference proofs should be transferred more selectively.

4 Internalizing Primitive-Guided Reasoning

In this section, we introduce ABSORB, a novel variant of on-policy self-distillation that improves the mathematical problem-solving capability of LLMs by internalizing primitive-guided reasoning, motivated by the observations in §3. Extensive experiments with Qwen models as backbones show that ABSORB consistently outperforms strong baselines on a diverse set of mathematical reasoning benchmarks (§4.4). Further implementation details and ablation results are provided in Appendix E.

4.1 Primitives as Privileged Information

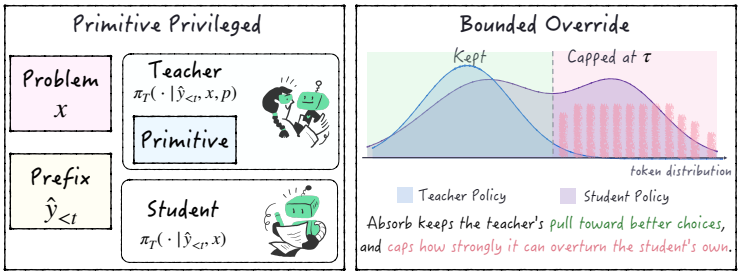


Figure 5: Illustration of ABSORB paradigm.

Our diagnosis in §3 identifies missing mathematical primitives as the primary bottleneck in discovery-limited problems. A natural remedy is to directly supervise primitive generation. In practice, however, this objective transfers poorly to solution reasoning, likely reflecting a mismatch between structural understanding and step-by-step generation, as further supported by the ablations in Appendix E. More importantly, §3.3 shows that post-training can repair discovery-limited failures while leaving Discovery largely unchanged. This dissociation suggests that effective reasoning does not require primitives to be explicitly generated; instead, they can guide the reasoning trajectory implicitly.

Motivated by this observation, we provide **mathematical primitives as privileged information** to the teacher rather than as prediction targets for the student. This mirrors our **Execution** evaluation, where providing the primitive unlocks latent execution capability, and uses the same primitive-conditioned reasoning to guide post-training. Compared with full reference proofs, primitives provide more targeted supervision by supplying the missing structural information without prescribing execution steps the student can already perform. This design transfers **primitive-guided reasoning** into the student's own generation process without requiring primitives at inference time.

4.2 Bounded Override

Given the reasoning drift observed with existing post-training methods, indiscriminate teacher intervention should be avoided. Moreover, introducing mathematical primitives as privileged information can cause substantial privilege leakage under naive OPSD on complex reasoning problems, as shown in Appendix E. This motivates a selective transfer mechanism that preserves useful privileged guidance while limiting how strongly it can override the student's own reasoning. Intuitively, the teacher should **pull the student toward better choices without abruptly displacing preferences** that the student has already formed. The primitive should refine the student's existing reasoning space rather than force it to imitate decisions that are only accessible under privileged context.

This leads to two design principles: ❶ when the privileged teacher assigns higher preference to a choice than the student, this positive guidance should be transferred in full, as it directly reflects information contributed by the primitive; and ❷ when the teacher strongly suppresses a choice favored by the student, the resulting pressure should be bounded, since large disagreement may arise from privileged information unavailable to the student rather than from an error it can independently recognize. Importantly, bounding this negative pressure does not freeze incorrect student preferences: positive guidance toward better alternatives remains fully active, progressively shifting probability mass away from suboptimal choices over training. These desiderata admit a simple implementation

Model	PRIM		HLE Math	HMMT25	Omni-MATH	Avg.
	Discovery	Generation				
Qwen3.5-4B	6.59	26.37	26.70	76.67	78.00	51.94
+ SFT	5.49 (-1.10)	25.27 (-1.10)	27.36 (+0.65)	83.33 (+6.66)	79.33 (+1.33)	53.82 (+1.89)
+ OPSD	8.79 (+2.20)	30.22 (+3.85)	29.58 (+2.88)	83.33 (+6.66)	79.33 (+1.33)	55.62 (+3.68)
+ ABSORB	**9.89** (+3.30)	**31.32** (+4.95)	**31.02** (+4.32)	83.33 (+6.66)	**80.00** (+2.00)	**56.42** (+4.48)
Qwen3.5-9B	13.19	38.46	35.60	90.00	78.67	60.68
+ SFT	12.64 (-0.55)	32.42 (-6.04)	34.55 (-1.05)	90.00 (+0.00)	80.67 (+2.00)	59.41 (-1.27)
+ OPSD	10.99 (-2.20)	33.52 (-4.95)	35.08 (-0.52)	90.00 (+0.00)	81.33 (+2.67)	59.98 (-0.70)
+ ABSORB	**14.29** (+1.10)	**43.96** (+5.49)	**38.35** (+2.75)	**93.33** (+3.33)	**82.00** (+3.33)	**64.41** (+3.73)
Qwen3.5-27B	**28.57**	47.80	42.67	96.67	87.33	68.62
+ SFT	28.02 (-0.55)	48.35 (+0.55)	43.59 (+0.92)	96.67 (+0.00)	86.67 (-0.67)	68.82 (+0.20)
+ OPSD	**28.57** (+0.00)	44.51 (-3.30)	42.80 (+0.13)	96.67 (+0.00)	88.00 (+0.67)	67.99 (-0.62)
+ ABSORB	28.02 (-0.55)	**48.90** (+1.10)	**46.60** (+3.93)	**100.00** (+3.33)	**88.67** (+1.33)	**71.04** (+2.42)

Table 4: Comparison of SFT, OPSD, and ABSORB across three Qwen3.5 model scales.

through reverse KL with one-sided clamping, which preserves positive teacher guidance while bounding excessive suppression of student-preferred choices.

4.3 OVERALL OBJECTIVE

Combining the two design principles above, we introduce ABSORB, illustrated in Figure 5. ABSORB conditions the teacher on the mathematical primitive p to provide targeted guidance for the missing discovery signal, while bounding how strongly this can override the student's own reasoning. Let $\mathcal{S}_t$ denote the teacher's top-K token support at step t, and let $\bar{\pi}_S$ and $\bar{\pi}_T$ denote the student and privileged-teacher distributions renormalized over $\mathcal{S}_t$, respectively. The resulting objective is:

$$\mathcal{L}_{\text{ABSORB}} = \frac{1}{T}\sum_{t=1}^{T}\sum_{v\in\mathcal{S}_t} \min\left\{\bar{\pi}_S(v \mid \hat{y}_{<t}, x)\log\frac{\bar{\pi}_S(v \mid \hat{y}_{<t}, x)}{\bar{\pi}_T(v \mid \hat{y}_{<t}, x, p)}, \tau\right\}.$$

Here, the privileged teacher injects primitive-guided information along the student's on-policy reasoning trajectory, while the one-sided clamp preserves its positive guidance toward better alternatives and bounds excessive suppression of choices favored by the student.

4.4 EXPERIMENTAL RESULTS

Setup. For implementing ABSORB, we utilize the identical base models and optimization configurations detailed in §3.3, differing only in the substitution of the standard OPSD objective with our primitive-privileged divergence loss ($\mathcal{L}_{\text{ABSORB}}$). To comprehensively evaluate internalized reasoning capabilities and generalization, we assess performance across four challenging mathematical benchmarks: our curated PRIM (evaluating both Generation and Discovery), the full mathematics subset of HLE-Verified (Zhai et al., 2026), HMMT25 (Dekoninck et al., 2026), and Omni-MATH (Gao et al., 2025). To account for the extreme difficulty of the competition-level datasets, we report pass@4 accuracy for HMMT25 and Omni-MATH. The overall average (Avg.) is computed exclusively over the Generation scores across all four benchmarks, excluding Discovery.

Results. As shown in Table 4, ABSORB consistently outperforms SFT and OPSD across model scales and benchmarks. At the 9B scale, SFT and OPSD reduce Generation accuracy by 6.04 and 4.95 percentage points, respectively, while OPSD incurs a 3.30-point drop at the 27B scale. These regressions explain the modest net gains discussed in §3.3: repairing previously failed problems can be offset by newly introduced errors on problems the model already solves. In contrast, ABSORB improves Generation at all three model scales, including a 5.49 point gain at 9B and a 3.73 point improvement in average benchmark performance. This pattern is consistent with the role of bounded override: selective transfer retains useful primitive-guided information while limiting disruptive changes to the model's existing reasoning behavior. The results also support the intended internalization of primitive-guided reasoning. Improvements from ABSORB are concentrated in downstream reasoning

rather than explicit Discovery. At the 9B scale, Generation improves by 5.49 points while Discovery increases by only 1.10; at 27B, Generation improves by 1.10 points and HLE Math by 3.93, despite a 0.55-point decrease in Discovery. Thus, the benefit of the primitive need not manifest as better explicit primitive recovery; instead, its structural guidance is increasingly reflected in the model's unassisted reasoning process.

5 Related Work

Mathematical Reasoning in LLMs. Mathematical reasoning benchmarks have rapidly escalated in difficulty, progressing from early grade-school and competition datasets (e.g., MathQA (Amini et al., 2019), ASDiv (Miao et al., 2020), SVAMP (Patel et al., 2021), GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021)) to advanced Olympiad- and expert-level frontiers (TheoremQA (Chen et al., 2023), MathBench (Liu et al., 2024), AIME (Dekoninck et al., 2026), OlympiadBench (He et al., 2024), Omni-MATH (Gao et al., 2025), HLE (Center for AI Safety et al., 2026), FrontierMath (Glazer et al., 2024)). Despite this scaling in complexity, evaluation predominantly remains rigidly anchored to final-answer correctness. Recognizing the limitations of outcome-based metrics, recent literature explores process-oriented evaluation: SMART (Hou et al., 2026) decomposes problem-solving across cognitive dimensions, ReasoningMath-Plus (Zheng et al., 2026) verifies minimal procedural skeletons, and DeepInsightTheorem (Li et al., 2026a) leverages proof sketches for intermediate supervision. Executable Functional Abstractions (EFAs) (Khan et al., 2025) represent mathematical structure as executable generative programs, whereas our Mathematical Primitive captures the concise, non-procedural idea underlying a specific solution. Our approach instead isolates the load-bearing *mathematical primitive*, decoupling structural understanding from procedural execution. We evaluate whether models can discover it from a problem, recover it from a solution, and execute the proof when it is provided.

On-Policy Self-Distillation. On-Policy Self-Distillation (OPSD) advances knowledge distillation for LLM reasoning by mitigating the distribution shift inherent in traditional off-policy methods (Zhao et al., 2026; He et al., 2026; Penaloza et al., 2026). Specifically, the student generates its own reasoning trajectories on-policy, while the teacher, conditioned on additional privileged information, provides dense supervision along these rollouts, with both sharing the same base model. However, recent studies show that naive OPSD can suffer from privilege leakage and suppress exploratory or self-corrective reasoning. To address this, Shen et al. (2026) isolate inference-transferable supervision from reference-induced shortcuts, while Kaur et al. (2026); Peng et al. (2026) focus on preserving high-entropy decision points and epistemic behaviors from aggressive privileged-teacher corrections. Other approaches modulate supervision based on teacher–student disagreement (Li et al., 2026b; Yu et al., 2026; Xie et al., 2026), or utilize RL primarily for token-level credit assignment (Yang et al., 2026). AVSD mitigates privilege leakage by separating shared and view-specific signals across multiple privileged views (Nguyen et al., 2026). ABSORB differs primarily in the privileged signal itself. Rather than conditioning the teacher on a full solution, ABSORB uses the primitive as privileged information and selectively transfers its guidance through a bounded override mechanism.

6 Conclusion

In this paper, we introduced the notion of Mathematical Primitive and PRIM, a benchmark for systematically evaluating structural mathematical understanding in LLMs across Discovery, Generation, Digestion, and Execution. Our diagnosis reveals that similar solution accuracy can mask markedly different capability profiles, correct primitives can unlock substantial latent execution capacity, and independent Discovery constitutes a dominant bottleneck in mathematical reasoning. We further show that discovery-limited failures are substantially more amenable to post-training, while existing methods can introduce regressions on problems the model already solves. Building on these findings, we proposed ABSORB, a primitive-privileged post-training paradigm that selectively transfers primitive-guided reasoning through a bounded override mechanism. Experiments on challenging mathematical reasoning benchmarks show that ABSORB consistently improves over strong post-training baselines. Overall, our results highlight the value of explicitly diagnosing structural mathematical understanding and leveraging this structure to improve LLM reasoning.

AI USE STATEMENT

In this work, we used generative AI tools to generate synthetic data sets, design or provide feedback on research methodology or experiments, implement methods, assist with translation, and clean and reformat dataset, edit a research paper to improve readability. We have not used generative AI tools to help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, support qualitative and thematic data analysis, or interpret results.

Additionally, we used generative AI tools to create or modify scientific figures or images and create or edit software code. We have reviewed all AI-assisted work. All LLM-generated code was thoroughly tested and verified by three authors. All formatted tables within the manuscript were manually cross-validated against the raw data by two authors to guarantee accuracy. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2357–2367, 2019.

Johan Boye and Birger Moell. Large language models and mathematical reasoning failures. *arXiv preprint arXiv:2502.11574*, 2025.

Sébastien Bubeck, Christian Coester, Ronen Eldan, Timothy Gowers, Yin Tat Lee, Alexandru Lupsasca, Mehtaab Sawhney, Robert Scherrer, Mark Sellke, Brian K. Spears, Derya Unutmaz, Kevin Weil, Steven Yin, and Nikita Zhivotovskiy. Early science acceleration experiments with GPT-5. *arXiv preprint arXiv:2511.16072*, 2025. URL https://arxiv.org/abs/2511.16072.

Center for AI Safety, Scale AI, and HLE Contributors Consortium. A benchmark of expert-level academic questions to assess AI capabilities. *Nature*, 649:1139–1146, 2026. doi:10.1038/s41586-025-09962-4. URL https://arxiv.org/abs/2501.14249.

Wenhu Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony Xia. Theoremqa: A theorem-driven question answering dataset. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7889–7901, 2023.

Claude. More than two thirds of the zeros of the riemann zeta function are simple and on the critical line. Anthropic manuscript, 2026. URL https://www-cdn.anthropic.com/95c246936988e43127bc6b2ceb7077c1dad2d68e.pdf.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Jasper Dekoninck, Nikola Jovanović, Tim Gehrunger, Kári Rögnvaldsson, Ivo Petrov, Chenhao Sun, and Martin Vechev. Beyond benchmarks: Matharena as an evaluation platform for mathematics with llms. 2026. URL https://arxiv.org/abs/2605.00674.

Simon Frieder, Jonas Bayer, Sam Looi, Jacob Loader, Julius Berner, Katherine M Collins, András Juhász, Fabian Ruehle, Sean Welleck, Gabriel Poesia, et al. Data for mathematical copilots: Better ways of presenting proofs for machine learning. *arXiv preprint arXiv:2412.15184*, 2024.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Zhengyang Tang, et al. Omni-math: A universal olympiad level mathematic benchmark for large language models. In *International Conference on Learning Representations*, volume 2025, pp. 100540–100569, 2025.

Elliot Glazer, Ege Erdil, Tamay Besiroglu, Diego Chicharro, Evan Chen, Alex Gunning, Caroline Falkman Olsson, Jean-Stanislas Denain, Anson Ho, Emily de Oliveira Santos, et al. Frontiermath: A benchmark for evaluating advanced mathematical reasoning in ai. *arXiv preprint arXiv:2411.04872*, 2024.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, 2024.

Yinghui He, Simran Kaur, Adithya Bhaskar, Yongjin Yang, Jiarui Liu, Narutatsu Ri, Liam Fowl, Abhishek Panigrahi, Danqi Chen, and Sanjeev Arora. Self-distillation zero: Self-revision turns binary rewards into dense supervision. *arXiv preprint arXiv:2604.12002*, 2026.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Yujie Hou, Mei Wang, Yaoyao Zhong, Ting Zhang, Xuetao Ma, and Hua Huang. SMART: Evaluating LLMs' mathematical reasoning via a human cognitive process-inspired benchmark. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 35426–35452, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi:10.18653/v1/2026.acl-long.1638. URL https://aclanthology.org/2026.acl-long.1638/.

Uijeong Jang and Ernest K. Ryu. Point convergence of nesterov's accelerated gradient method: An ai-assisted proof. *arXiv preprint arXiv:2510.23513*, 2025. URL https://arxiv.org/abs/2510.23513.

Simran Kaur, Narutatsu Ri, Yinghui He, Liam Fowl, and Sanjeev Arora. Rethinking on-policy self-distillation for thinking models. *arXiv preprint arXiv:2607.05184*, 2026.

Zaid Khan, Elias Stengel-Eskin, Archiki Prasad, Jaemin Cho, and Mohit Bansal. Executable functional abstractions: Inferring generative programs for advanced math problems. *arXiv preprint arXiv:2504.09763*, 2025.

Yunhe Li, Hao Shi, Bowen Deng, Wei Wang, Mengzhe Ruan, Hanxu Hou, Zhongxiang Dai, Siyang Gao, Chao Wang, Shuang Qiu, et al. Learning to reason with insight for informal theorem proving. *arXiv preprint arXiv:2604.16278*, 2026a.

Yunhe Li, Hao Shi, Wenhao Liu, Mengzhe Ruan, Hanxu Hou, Zhongxiang Dai, Shuang Qiu, and Linqi Song. Demopsd: Disagreement-modulated policy self-distillation. *arXiv preprint arXiv:2607.02502*, 2026b.

Hongwei Liu, Zilong Zheng, Yuxuan Qiao, Haodong Duan, Zhiwei Fei, Fengzhe Zhou, Wenwei Zhang, Songyang Zhang, Dahua Lin, and Kai Chen. Mathbench: Evaluating the theory and application proficiency of llms with a hierarchical mathematics benchmark. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 6884–6915, 2024.

Minghai Lu, Benjamin Delaware, and Tianyi Zhang. Proof automation with large language models. In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*, pp. 1509–1520, 2024.

Shen-yun Miao, Chao-Chun Liang, and Keh-Yih Su. A diverse corpus for evaluating and developing English math word problem solvers. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 975–984, Online, July 2020. Association for Computational Linguistics. doi:10.18653/v1/2020.acl-main.92. URL https://aclanthology.org/2020.acl-main.92/.

Duy Nguyen, Hanqi Xiao, Archiki Prasad, Zaid Khan, Anirban Das, Austin Zhang, Sambit Sahu, Hyunji Lee, Elias Stengel-Eskin, and Mohit Bansal. Avsd: Adaptive-view self-distillation by balancing consensus and teacher-specific privileged signals. *arXiv preprint arXiv:2605.20643*, 2026.

OpenAI. gpt-oss-120b & gpt-oss-20b model card, 2025. URL https://arxiv.org/abs/2508.10925.

OpenAI. Introducing gpt-5.4, March 2026a. URL https://openai.com/index/introducing-gpt-5-4/.

OpenAI. Introducing gpt-5.4 mini and nano, March 2026b. URL https://openai.com/index/introducing-gpt-5-4-mini-and-nano/.

OpenAI. Ten advances in mathematics and theoretical computer science. https://openai.com/index/ten-advances-in-mathematics/, 2026a. Astra Model Research Release and Lean 4 Formalizations.

OpenAI. Planar point sets with many unit distances. Manuscript, 2026b. URL https://cdn.openai.com/pdf/74c24085-19b0-4534-9c90-465b8e29ad73/unit-distance-proof.pdf.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are nlp models really able to solve simple math word problems? In *Proceedings of the 2021 conference of the North American chapter of the association for computational linguistics: human language technologies*, pp. 2080–2094, 2021.

Emiliano Penaloza, Dheeraj Vattikonda, Nicolas Gontier, Alexandre Lacoste, Laurent Charlin, and Massimo Caccia. Privileged information distillation for language models. *arXiv preprint arXiv:2602.04942*, 2026.

Keqin Peng, Chen Li, Yuanxin Ouyang, Yancheng Yuan, and Liang Ding. Diagnosing and mitigating thinking collapse in on-policy self-distillation. *arXiv preprint arXiv:2607.10805*, 2026.

Tian Qin, Core Francisco Park, Mujin Kwun, Aaron Walsman, Eran Malach, Nikhil Anand, Hidenori Tanaka, and David Alvarez-Melis. Decomposing elements of problem solving: What" math" does rl teach? *arXiv preprint arXiv:2505.22756*, 2025.

Qwen Team. Qwen3.5: Accelerating productivity with native multimodal agents, February 2026a. URL https://qwen.ai/blog?id=qwen3.5.

Qwen Team. Qwen3.6-27B: Flagship-level coding in a 27b dense model, April 2026b. URL https://qwen.ai/blog?id=qwen3.6-27b.

Zhanming Shen, Jintao Tong, Shaotian Yan, Chen Shen, Hao Chen, Wentao Ye, Xiaomeng Hu, Rui Miao, Haobo Wang, Junbo Zhao, et al. Purified opsd: On-policy self-distillation without losing how to think. *arXiv preprint arXiv:2607.02234*, 2026.

Terence Tao. Mathematics in the age of ai. *arXiv preprint arXiv:2608.16753*, 2026.

Zhengpeng Xie, Li Lyna Zhang, Zeke Xie, and Mao Yang. Trust region policy distillation. *arXiv preprint arXiv:2607.04751*, 2026.

Chenxu Yang, Chuanyu Qin, Qingyi Si, Minghui Chen, Naibin Gu, Dingyu Yao, Zheng Lin, Weiping Wang, Jiaqi Wang, and Nan Duan. Self-distilled rlvr. *arXiv preprint arXiv:2604.03128*, 2026.

Zichao Yu, Chengzhi Yu, Shengze Xu, Yujin Han, Bingqing Jiang, Xu Wang, and Difan Zou. Mismatch matters: On-policy distillation beyond token agreement. *arXiv preprint arXiv:2608.09836*, 2026.

Weiqi Zhai, Zhihai Wang, Jinghang Wang, Boyu Yang, Xiaogang Li, Xander Xu, Bohan Wang, Peng Wang, Xingzhe Wu, Anfeng Li, et al. Hle-verified: A systematic verification and structured revision of humanity's last exam. *arXiv preprint arXiv:2602.13964*, 2026.

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. *arXiv preprint arXiv:2601.18734*, 2026.

Xiang Zheng, Weiqi Zhai, Wei Wang, Boyu Yang, Wenbo Li, Ruixiang Luo, Haoxiang Sun, Yucheng Wang, Zhengze Li, Meng Wang, et al. Unmasking reasoning processes: A process-aware benchmark for evaluating structural mathematical reasoning in llms. *arXiv preprint arXiv:2602.00564*, 2026.

A Data Curation Details

A.1 Dataset Filtering and Correction

To reduce annotation noise in the original HLE, we draw from the community-verified HLE-Verified release (Zhai et al., 2026), retaining only its *Gold* and *Revision* subsets. We randomly sample 200 text-only problems and conduct an additional expert review for primitive-level evaluation. This process excludes 18 unsuitable problems due to ill-posed or ambiguous questions, incorrect or unverifiable answers, non-self-contained tasks, insufficient rationales, exhaustive-computation requirements, or other quality issues. For 17 of these 18 problems, no valid gold primitive could be established, yielding a final benchmark of 182 problems.

Expert review further identifies residual errors in the official answers of 15 retained problems, including 13 mathematical errors and 2 corrupted answer strings, which we correct in our benchmark. Re-evaluating the strongest model on these corrected answers flips 11 predictions from incorrect to correct, with no reverse flips, providing an additional consistency check on the revisions.

Because gold primitives are derived with access to reference solutions, we also audit them for answer leakage. We search the concatenated primitive fields for bounded occurrences of the gold answer, using token boundaries designed to avoid spurious numeral matches. This audit initially identifies 4 affected primitives. We manually revise them to remove the leaked answers while preserving their structural content, and verify that no direct answer-string leakage remains in the final benchmark.

A.2 Primitive Type and Family

To make the definition operational and to structure evaluation, we anchor every primitive to a frozen taxonomy of ten primitive *types* (plus a `computation_only` escape hatch for problems that genuinely have no kernel primitive in the sense of Definition 1), as presented in Table 5.

Type	**The essential property is…**
`reformulation`	a change of language/encoding/viewpoint makes the problem tractable
`hidden_structure`	the objects secretly carry a known structure (group, graph, metric space, …)
`invariant_or_monotonicity`	a preserved or monotone quantity constrains all transformations
`theorem_applicability`	the task reduces to verifying the hypotheses of a known theorem
`duality_or_representation`	a dual / spectral / transformed viewpoint simplifies the problem
`extremal_principle`	a maximal/minimal/extreme object forces structure
`construction_or_witness`	an explicit object/certificate/counterexample is built
`reduction_or_embedding`	the problem is reduced/embedded into a known canonical one
`obstruction_or_contradiction`	the opposite assumption violates a necessary constraint
`induction_or_recursion`	a recursive/self-similar structure with a strengthened hypothesis

Table 5: The ten primitive types.

Further, since the same problem can often be solved through conceptually different routes, we further assign the primary primitive type as presented in Table 6 of each problem to one of three *families*: Recast, Witness, or Argument, as summarized below:

- **Recast**: The load-bearing object is the *target setting* Y into which the original problem is moved, such as an equivalent reformulation, a hidden known structure, a canonical target problem, a dual or representational viewpoint, or a named theorem. The gate asks whether the model recasts the problem into the same Y, or a provably equivalent one up to trivial restatement. The mechanism asks how this target setting unlocks the solution, such as the reduction step actually used, or, in theorem-based cases, whether the theorem's hypotheses are correctly verified. Moving to a different setting, naming the wrong theorem, or remaining entirely within the original framing constitutes a gate mismatch; identifying a coarser or weaker target receives partial credit.
- **Witness**: The load-bearing object is a specific mathematical *object*: a construction, certificate, witness, extremal object, invariant, or monotone quantity. The gate asks whether the model identifies the same object, or an equivalent object that plays the identical role in the solution. The

Family	Routed primitive types	Gate (WHAT)	Mechanism (HOW)
RECAST	`reformulation`, `hidden_structure`, `reduction_or_embedding`, `duality_or_representation`, `theorem_applicability`	target setting Y (equivalent form / hidden structure / canonical target / dual / named theorem)	how Y unlocks it; for theorems, hypotheses verified
WITNESS	`construction_or_witness`, `extremal_principle`, `invariant_or_monotonicity`	the object (construction / certificate / extremal / invariant–monotone)	its decisive property: built to satisfy / extremality / invariance forces it
ARGUMENT	`obstruction_or_contradiction`, `induction_or_recursion`	the load-bearing claim (specific obstruction / strengthened hypothesis)	why the opposite fails / how the recursion closes

Table 6: The primitive families. Each of the ten frozen primitive types routes to exactly one family.

mechanism asks for its decisive property: for example, why the construction satisfies the required constraint, why extremality forces the conclusion, or why invariance forbids or forces the desired outcome. An object of the correct general kind but without the decisive feature receives partial credit, such as a near-construction that fails the needed property or a monotone quantity that is not the one driving the proof.

- **ARGUMENT**: The load-bearing object is the central *claim* on which the proof depends, such as the obstruction contradicted by the opposite assumption or the strengthened induction hypothesis needed for the recursive step. The gate asks whether the model's reasoning hinges on the same claim; merely naming the proof style, such as "by contradiction" or "by induction", is not sufficient. The mechanism asks for the logical step itself, such as why the opposite assumption violates the constraint or how the strengthened hypothesis closes the induction. Stating only the proof style without the specific load-bearing claim is a gate mismatch, while a related but loosely stated or under-strengthened claim receives partial credit.

A.3 PRIMITIVE SCORING PROTOCOL

Mathematical primitive correctness is evaluated against the annotated *gold primitive*, with the problem's full reference solution (*rationale*) supplied as supporting context. The rationale is provided only so the judge can recognize coarser phrasings, equivalent paraphrases, or alternative routes that the reference solution shows to be valid, i.e. so a correct primitive is not penalized merely for failing to match the terse gold annotation verbatim, while never requiring the model to reproduce the downstream execution.

All three primitive families share the same three-verdict decomposition $(V, \sigma_{\text{gate}}, \sigma_{\text{mech}})$ and the deterministic scoring rule in § 2.3. Here, $V \in \{0, 1\}$ is a binary validity indicator (is the output a genuine primitive rather than a generic remark, a bare technique name, or pure computation), while the two graded slots $\sigma_{\text{gate}}, \sigma_{\text{mech}} \in \{0, \frac{1}{2}, 1\}$ evaluate whether the response identifies the essential WHAT (the *gate*) and explains the operative HOW (the *mechanism*). The difference across families is the type of mathematical object over which these two slots are instantiated, ensuring that "the same idea" is matched against the appropriate target. Because V is binary and $\sigma_{\text{gate}}, \sigma_{\text{mech}} \in \{0, \frac{1}{2}, 1\}$, the score lies in $[0, 1]$ but is restricted to the finite set, and we have $\text{Score} \in \{0,\ 0.30,\ 0.40,\ 0.50,\ 0.60,\ 0.80,\ 1.00\}$.

All primitive judgments are produced by GPT-5.4-High at temperature 0. To validate the judge, we sampled 200 (model, problem) pairs, 100 judged primitive-correct and 100 judged primitive-incorrect, stratified across the 12 evaluated models. Three annotators with graduate-level mathematical training, blind to the judge's verdict, independently labeled PrimitiveAcc. Judge–human agreement is 95.5% (Cohen's $\kappa = 0.91$), with 9 disagreements.

Because each problem is scored against a single gold primitive, a response following a genuinely different but valid conceptual route may be under-credited. To assess this limitation, we further examined the 100 primitive-incorrect cases in the same validation sample (8–9 per model, fixed seed): exactly 1 case (1%) contained a valid alternative primitive missed by the single-gold rubric. We additionally audited all 32 problems marked primitive-incorrect for the strongest model (gpt-5.4, $\mathrm{Score} < 0.8$), identifying exactly 1 such case. These audits suggest that, although the single-gold rubric may conservatively underestimate valid primitive generation, under-counting due to alternative conceptual routes is rare in the audited cases.

A.4 Illustrative Example of Mathematical Primitives

To make the notion of a mathematical primitive concrete, we provide a representative example in Figure 6. The full rationale contains the complete derivation needed to solve the problem, whereas the primitive isolates the underlying structural idea that makes the derivation possible. Due to space constraints, we show only the `core_concept` field in the figure. This example illustrates that a primitive is not a shortened proof, but a compact representation of the key mathematical structure underlying the solution.

A.5 Prompt on Extracting a Mathematical Primitive and Primitive Scoring

During data curation, we use a teacher model to extract a mathematical primitive for each problem. As shown in Figure 7, the teacher is asked to identify the essential mathematical property underlying the problem and explain the solution principle enabled by that property, while explicitly avoiding calculations or solution steps. We additionally require a structured primitive classification, including a primary type, optional secondary types, a classification rationale, and a confidence score.

We evaluate each predicted primitive using three complementary signals. First, the validity term V checks whether the prediction contains a substantive mathematical primitive rather than a vacuous response, such as generic advice, a bare technique name, or computation without a conceptual claim (Figure 8). For predictions that pass this check, we separately assess the correctness of the GATE and MECHANISM components using the reference solution and gold primitive (Figure 9). The GATE score σ_{gate} measures whether the prediction identifies the essential problem-specific idea, the "what" that unlocks the problem, while the MECHANISM score σ_{mech} measures whether it correctly explains the high-level "how," namely why that idea enables a valid solution principle. Both correctness judgments emphasize semantic agreement rather than exact reproduction and do not penalize missing downstream execution details.

B Additional Details on Diagnostic Evaluation

B.1 Comparison with Full-HLE Performance

Notably, the absolute answer accuracies in Table 7 are substantially higher than the commonly reported performance of these model families on the full HLE benchmark. This difference should not be interpreted as a contradiction, for two reasons: ❶ PRIM is a curated scorable subset rather than the full HLE distribution; and ❷ during benchmark construction, human experts removed examples with annotation noise or evaluation ambiguity in the original HLE (Center for AI Safety et al., 2026) and HLE-Verified (Zhai et al., 2026) sources, including fundamentally incorrect reference answers, logical gaps in rationales, and irregular formatting that hinders automated evaluation.

B.2 Prompt on Generating the Primitive and Plan using GPT-5.4

We use GPT-5.4 to generate two forms of teacher guidance for the Execution analysis: a Teacher Primitive, which captures the load-bearing mathematical structure following Definition 1, and a Teacher Plan, which provides a step-by-step solution outline. The prompts used to generate these two forms of guidance are provided below in Figures 10 and 11.

Question

Let

$$L := \mathbb{Q}\left(\sqrt{(2+\sqrt{2})(3+\sqrt{3})}, \sqrt{2}, \sqrt{3}\right).$$

What is the Galois group of $L/\mathbb{Q}$?

Rationale

Set $\alpha = \sqrt{2}$, $\beta = \sqrt{3}$, and

$$\gamma = \sqrt{(2+\alpha)(3+\beta)}.$$

Let $K = \mathbb{Q}(\alpha, \beta)$. Then $K/\mathbb{Q}$ is biquadratic, hence Galois with $[K : \mathbb{Q}] = 4$ and $\mathrm{Gal}(K/\mathbb{Q}) \cong C_2 \times C_2$.

Step 1: $[L : K] = 2$ and $[L : \mathbb{Q}] = 8$
We have $\gamma^2 = (2+\alpha)(3+\beta) \in K$, so $L = K(\gamma)$ and $[L : K] \leq 2$. To see $\gamma \notin K$, it suffices to show $a := (2+\alpha)(3+\beta)$ is not a square in K.
Consider the relative norm $N_{K/\mathbb{Q}(\alpha)}$. The nontrivial automorphism of $K/\mathbb{Q}(\alpha)$ sends $\beta \mapsto -\beta$, hence

$$N_{K/\mathbb{Q}(\alpha)}(a) = a \cdot a^{\tau} = (2+\alpha)(3+\beta)\cdot(2+\alpha)(3-\beta) = (2+\alpha)^2(9-\beta^2) = 6(2+\alpha)^2.$$

If a were a square in K, then its norm down to $\mathbb{Q}(\alpha)$ would be a square in $\mathbb{Q}(\alpha)$. But $6(2+\alpha)^2$ is a square in $\mathbb{Q}(\alpha)$ if and only if 6 is a square in $\mathbb{Q}(\alpha)$. Now 6 is not a square in $\mathbb{Q}(\sqrt{2})$: if $(u+v\alpha)^2 = 6$ with $u, v \in \mathbb{Q}$, then the α-coefficient gives $2uv = 0$, so either $v = 0$ (forcing $u^2 = 6$, impossible) or $u = 0$ (forcing $2v^2 = 6$, i.e. $v^2 = 3$, impossible in $\mathbb{Q}$).
Hence a is not a square in K, so $[L : K] = 2$ and therefore $[L : \mathbb{Q}] = 8$.

Step 2: Explicit automorphisms and normality
Define the involution

$$\rho : \ \alpha \mapsto \alpha, \ \beta \mapsto \beta, \ \gamma \mapsto -\gamma.$$

This generates $\mathrm{Gal}(L/K) \cong C_2$.
Next extend the two generators of $\mathrm{Gal}(K/\mathbb{Q})$. Let σ_0 be the automorphism of K with $\alpha \mapsto -\alpha, \ \beta \mapsto \beta$. Note that

$$\sigma_0(\gamma^2) = (2-\alpha)(3+\beta).$$

But

$$\frac{(2-\alpha)(3+\beta)}{(2+\alpha)(3+\beta)} = \frac{2-\alpha}{2+\alpha} = 3 - 2\alpha = (\alpha-1)^2,$$

so $\sigma_0(\gamma^2) = (\alpha-1)^2\gamma^2$. Hence we can define an extension $\sigma \in \mathrm{Aut}(L)$ by

$$\sigma : \ \alpha \mapsto -\alpha, \ \beta \mapsto \beta, \ \gamma \mapsto (\alpha-1)\gamma,$$

since then $\sigma(\gamma)^2 = (\alpha-1)^2\gamma^2 = \sigma_0(\gamma^2)$.
Similarly let τ_0 be the automorphism of K with $\alpha \mapsto \alpha, \ \beta \mapsto -\beta$. Then

$$\tau_0(\gamma^2) = (2+\alpha)(3-\beta), \quad \text{and} \quad \frac{(2+\alpha)(3-\beta)}{(2+\alpha)(3+\beta)} = \frac{3-\beta}{3+\beta} = 2 - \beta = \left(\frac{\beta-1}{\alpha}\right)^2$$

(using $(\beta-1)^2 = 4 - 2\beta$ and $\alpha^2 = 2$). Thus $\tau_0(\gamma^2) = \left(\frac{\beta-1}{\alpha}\right)^2\gamma^2$, and we define $\tau \in \mathrm{Aut}(L)$ by

$$\tau : \ \alpha \mapsto \alpha, \ \beta \mapsto -\beta, \ \gamma \mapsto \frac{\beta-1}{\alpha}\,\gamma.$$

These maps show every sign-change on α, β extends to L, so $L/\mathbb{Q}$ is Galois. Since $[L : \mathbb{Q}] = 8$, we now have $|\mathrm{Gal}(L/\mathbb{Q})| = 8$.

Step 3: Group structure
Compute squares on γ:

$$\sigma^2(\gamma) = \sigma\big((\alpha-1)\gamma\big) = \sigma(\alpha-1)\,\sigma(\gamma) = (-\alpha-1)(\alpha-1)\gamma = -(\alpha^2-1)\gamma = -\gamma,$$

so $\sigma^2 = \rho$. Likewise,

$$\tau^2(\gamma) = \tau\left(\frac{\beta-1}{\alpha}\gamma\right) = \tau\left(\frac{\beta-1}{\alpha}\right)\tau(\gamma) = \left(\frac{-\beta-1}{\alpha}\right)\left(\frac{\beta-1}{\alpha}\right)\gamma = -\frac{\beta^2-1}{\alpha^2}\gamma = -\frac{2}{2}\gamma = -\gamma,$$

so $\tau^2 = \rho$. In particular, σ and τ have order 4, and ρ is central of order 2.
Finally compare $\sigma\tau$ and $\tau\sigma$ on γ. Let $u = \alpha - 1$ and $v = (\beta-1)/\alpha$. Then $\sigma(v) = -v$ (because $\sigma(\alpha) = -\alpha$ and $\sigma(\beta) = \beta$), while $\tau(u) = u$. Hence

$$\sigma\tau(\gamma) = \sigma(v\gamma) = \sigma(v)\sigma(\gamma) = (-v)\,u\gamma = -(uv)\gamma, \quad \text{but} \quad \tau\sigma(\gamma) = \tau(u\gamma) = \tau(u)\tau(\gamma) = u\,v\gamma = (uv)\gamma.$$

Therefore $\sigma\tau(\gamma) = -\tau\sigma(\gamma)$, i.e. $\sigma\tau = \rho\,\tau\sigma$. So the group is generated by σ, τ with relations

$$\sigma^2 = \tau^2 = \rho, \quad \rho^2 = 1, \quad \rho \text{ central}, \quad \sigma\tau = \rho\,\tau\sigma,$$

which is exactly the quaternion group of order 8. Equivalently, every nontrivial element other than ρ has order 4, so the group cannot be dihedral; it is Q_8.

Primitive

core_concept. The key insight is to track the added radical by its square class over the biquadratic base; preservation of that square class gives the lifts, and the orders of those lifts together with whether they commute are what pin down the isomorphism type.

Figure 6: An example illustrating the distinction between a full solution rationale and the corresponding mathematical primitive.

You are given a language-only mathematics problem, its answer, and its rationale. Your task is to extract and classify the mathematical PRIMITIVE behind the rationale.

Definition:

A mathematical primitive is the essential mathematical property that makes the problem solvable, together with an explanation of how this property connects the problem to a valid solution principle. The primitive is not merely a theorem name, a technique, or a proof step. For example, "use induction," "apply Riesz representation," or "use generating functions" is insufficient unless it explains what property of the problem makes that method applicable.

Steps:

Step 1: Identify the essential mathematical property.

Ask: What hidden structure, invariant, relation, decomposition, obstruction, or theorem condition makes the problem tractable?

Step 2: Identify the solution principle.

Ask: How does that property enable a valid theorem, reduction, construction, reformulation, contradiction, or proof strategy? Name the essential conceptual move(s) -- one OR several distinct key ideas, whichever the problem genuinely needs.

Step 3: Write the mathematical primitive.

Write 1-3 sentences combining the essential property and its connection to the solution principle.

CRITICAL CONSTRAINTS:

- Strictly conceptual: no algebraic manipulation, equations, or step-by-step proof.
- HIGH-LEVEL but COMPLETE: state EVERY essential CONCEPTUAL idea the solution rests on (a problem may need ONE or SEVERAL distinct key observations -- include each one). Do NOT write out calculations, derivations, or intermediate numeric results.

Step 4: Classify the primitive.

Choose exactly one primary primitive type and zero to three secondary primitive types.

CRITICAL CONSTRAINT: You MUST use the exact string keys provided in the taxonomy below (e.g., "reformulation", "hidden_structure"). Do NOT modify, capitalize, or invent new types.

The primary primitive type should describe the main mathematical mechanism that unlocks the solution. The secondary primitive types should describe additional mathematical ideas that support, enable, or refine the primary primitive.

Do not classify based only on surface-level technique names. Classify based on the essential mathematical property and how it connects to the solution principle.

Primitive Taxonomy:

- **reformulation**: The problem becomes tractable by rewriting it in a more useful mathematical language, representation, or equivalent form.
- **hidden_structure**: The key is recognizing that the objects secretly have a known mathematical structure, such as a group, vector space, Hilbert space, graph, lattice, convex set, metric space, probability space, or algebraic structure.
- **invariant_or_monotonicity**: The key is identifying a quantity or property that is preserved, monotone, or constrains all valid transformations.
- **theorem_applicability**: The key is recognizing that the problem reduces to verifying the assumptions of a known theorem.
- **duality_or_representation**: The key is passing to a dual, transformed, spectral, functional, or representational viewpoint where the problem becomes simpler.
- **extremal_principle**: The key is choosing a maximal, minimal, largest, smallest, boundary, or otherwise extreme object and exploiting its extremality.
- **construction_or_witness**: The key is explicitly constructing an object, example, counterexample, certificate, function, sequence, graph, or witness.
- **reduction_or_embedding**: The key is reducing the problem to another known or canonical problem, or embedding it into a setting where a known result applies.
- **obstruction_or_contradiction**: The key is identifying a necessary structural constraint that the opposite assumption would violate.
- **induction_or_recursion**: The key is recognizing a recursive, self-similar, or inductive structure and choosing an induction hypothesis that preserves the essential property.

DO NOT SOLVE OR SUMMARIZE THE SOLUTION:

- Do NOT state or compute the final answer; no specific numeric result, final value, or boxed expression.
- Do NOT write a full derivation or proof.
- Do NOT write out the calculations or the step-by-step derivation. State all the essential conceptual primitive(s) the solution rests on -- including each distinct key idea it genuinely needs -- but not the execution that follows.

Extract and classify the mathematical primitive behind the rationale.

Output Format:

Output ONLY a valid JSON object in the exact format below, with no markdown code blocks formatting (e.g., do not wrap in ` ` ` json), just the raw JSON:

```
{
"essential_property": "...",
"solution_principle": "...",
"core_concept": "...",
"primary_primitive_type": "...",
"secondary_primitive_types": ["..."],
"classification_rationale": "...",
"confidence": <integer between 1 and 5>
}
```

Problem:

[PROBLEM]

Answer:

[GOLD_ANSWER]

Rationale:

[GOLD_RATIONALE]

Figure 7: Prompt used during data curation to generate mathematical primitives and their structured annotations.

You are checking whether a candidate mathematical "primitive" for a mathematics problem is VACUOUS -- i.e., not really a primitive at all. Look only at the candidate below.
A mathematical primitive names the essential property that makes the problem solvable and explains how it connects to a solution principle.
Mark verdict = "invalid" ONLY if the candidate is VACUOUS, i.e., ANY of:

- **"generic"**: vague advice with no problem-specific content ("use a clever substitution", "find the pattern").
- **"technique_name_only"**: names a method/theorem but never states the property of THIS problem that makes it apply.
- **"computation_only"**: only arithmetic/numeric work, with no conceptual claim.

Otherwise verdict = "valid", failure = "none".
Do NOT mark invalid for stating a conclusion or a final result, and do NOT mark invalid for being a full derivation -- over-completeness is NOT vacuity. A thorough or conclusion-stating answer can still contain a valid primitive.
Candidate Primitive:
essential_property: [MODEL essential_property]
solution_principle: [MODEL solution_principle]
core_concept: [MODEL core_concept]
Output Format:
Return JSON only:
{"verdict": "valid"|"invalid", "failure": "none"|"generic"|"technique_name_only"|"computation_only", "evidence": "<one short sentence>"}

Figure 8: Prompt for assessing primitive validity V by detecting vacuous predictions.

You are judging whether a MODEL's candidate mathematical primitive for a math problem is CORRECT on ONE dimension, using a known-correct REFERENCE SOLUTION. Judge correctness FOR THIS PROBLEM -- not reproduction of any particular wording.
GROUND RULES (read carefully):

- The REFERENCE SOLUTION below is correct and complete. The GOLD PRIMITIVE is a terse distillation of its key idea, written WITH the full solution in hand. The MODEL only saw the problem.
- Judge whether the MODEL's primitive correctly captures the [DIM] that unlocks this problem, as evidenced by the reference solution.
- CREDIT a correct primitive even if it is (i) phrased more coarsely or less precisely than the gold primitive, or (ii) a different but valid route that the reference solution shows would work.
- Do NOT require the model to reproduce the gold primitive's exact framing, or to name the same theorem if its statement is equivalent.
- Do NOT re-solve from scratch; judge the model's primitive against the reference solution.
- Do NOT consider whether the model reached any final answer -- judge the primitive content only.
- A primitive here is the SINGLE CORE high-level idea (the dominant "aha"). Everything DOWNSTREAM of it is execution, NOT part of the primitive, and its absence must NOT lower the score: secondary lemmas, second-step ideas, lower-bound / sharpness / optimality arguments, exact constants, specific numeric values, and calculation steps are what the core primitive ENABLES. Judge only whether the model has the correct CORE idea. (A vague, non-problem-specific statement, however, is still not a primitive.)

The dimension you are judging is the **[DIM]**:
[DIM_DEF]
Scoring (CORRECTNESS, not reproduction):

- **"match"**: [C_MATCH]
- **"partial"**: [C_PARTIAL]
- **"mismatch"**: [C_MISMATCH]

Problem:
[PROBLEM]
Reference Solution (correct):
[REFERENCE SOLUTION]
Gold Primitive (terse pointer to the essential idea):
essential_property: [GOLD essential_property]
solution_principle: [GOLD solution_principle]
core_concept: [GOLD core_concept]
Model Primitive (judge this):
essential_property: [MODEL essential_property]
solution_principle: [MODEL solution_principle]
core_concept: [MODEL core_concept]
Output Format:
Return JSON only:
{"agreement": "match"|"partial"|"mismatch", "evidence": "<one short sentence>"}

Figure 9: Prompt for assessing primitive correctness along the GATE and MECHANISM dimensions, corresponding to σ_{gate} and σ_{mech}.

```
You are given a language-only mathematics problem -- the problem ONLY, with no solution.  Your task
is to produce the single kernel mathematical PRIMITIVE needed to solve it:  the essential structural
understanding that makes the problem tractable.  Do NOT solve the problem.
## Definition:
A mathematical primitive is the essential mathematical structure or property that makes the problem
solvable, together with an explanation of how this structure connects the problem to a valid solution
principle.
The primitive is not merely a theorem name, a technique, or a proof step.  For example, "use induction,"
"apply Riesz representation," or "use generating functions" is insufficient unless it explains what
property of the problem makes that method applicable.
## Steps:
Step 1:  Identify the essential mathematical property.
Ask:  What hidden structure, invariant, relation, decomposition, obstruction, or theorem condition makes
the problem tractable?
Step 2:  Identify the solution principle.
Ask:  How does that property enable a valid theorem, reduction, construction, reformulation,
contradiction, or proof strategy?  Name the essential conceptual move(s) -- one OR several distinct
key ideas, whichever the problem genuinely needs.
Step 3:  Write the mathematical primitive.
Write 1-3 sentences capturing the single high-level "Aha!"  -- the central structural understanding that
makes the problem tractable.
## CRITICAL CONSTRAINTS:
 • Strictly conceptual:  no algebraic manipulation, equations, or step-by-step proof.
 • HIGH-LEVEL but COMPLETE: state EVERY essential CONCEPTUAL idea the solution rests on (a problem may
   need ONE or SEVERAL distinct key observations -- include each one).  Do NOT write out calculations,
   derivations, or intermediate numeric results.
 • Contrast:  "recognize the configuration is secretly a Cayley graph" is a mathematical primitive;
   "build the Cayley graph, compute its spectrum, then sum the eigenvalues to get the count" is a
   solution outline -- produce the former, never the latter.
## DO NOT SOLVE OR SUMMARIZE THE SOLUTION:
 • Do NOT state or compute the final answer; no specific numeric result, final value, or boxed
   expression.
 • Do NOT write a full derivation or proof.
 • Do NOT write out the calculations or the step-by-step derivation.  State all the essential conceptual
   primitive(s) the solution rests on -- including each distinct key idea it genuinely needs -- but not
   the execution that follows.
Produce the essential mathematical primitive(s):  the key thing(s) to notice before starting to solve.
## Output Format:
Output ONLY a valid JSON object in the exact format below, with no markdown code blocks formatting (e.g.,
do not wrap in ` ` ` json), just the raw JSON:
{
"essential_property":  "...",
"solution_principle":  "...",
"core_concept":  "..."
}
## Problem:
[PROBLEM]
```

Figure 10: Prompt for generating teacher mathematical primitives.

```
You are a math problem solver.  Below is a problem.  Produce a concise high-level solution plan
describing the major procedural steps needed to solve it.
## HARD CONSTRAINTS:
 • Give 3-6 major steps in the order they should be carried out.
 • Each step should state the intended mathematical operation, construction, theorem application, or
   reduction, without carrying out the corresponding derivation.
 • Do NOT provide algebraic calculations, intermediate numerical results, detailed derivations, or the
   final answer.
 • Keep the overall level of detail concise and comparable to a short conceptual guidance.
## Problem:
[PROBLEM]
```

Figure 11: Prompt for generating teacher solution plans from language-only mathematics problems.

B.3 Additional Experiment Results

B.3.1 Performance across Primitive Families

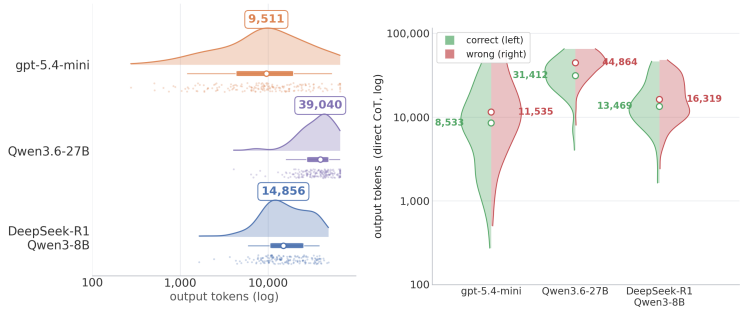
To provide a more fine-grained view of model behavior, we further break down performance by primitive family. Table 7 reports results on PRIM separately for **Recast**, **Witness**, and **Argument** primitives across all four dimensions of mathematical understanding. This breakdown complements the aggregate results in Table 1 in the main text by showing how the observed capability gaps vary across different forms of structural reasoning.

	Discovery			Generation			Digestion			Execution		
Model	**R**	**W**	**A**	**R**	**W**	**A**	**R**	**W**	**A**	**R**	**W**	**A**
OpenAI												
gpt-5.4	84.97	72.73	71.43	68.63	54.55	71.43	100.00	100.00	100.00	86.27	95.45	71.43
gpt-5.4-mini	61.44	54.55	28.57	51.63	40.91	57.14	97.39	100.00	100.00	70.59	81.82	71.43
gpt-5.4-nano	43.79	27.27	42.86	46.41	31.82	42.86	96.73	100.00	100.00	69.28	72.73	42.86
gpt-oss-20b	36.60	13.64	57.14	42.48	40.91	71.43	91.50	90.91	85.71	60.99	72.73	28.57
Qwen												
Qwen3.6-27B	28.76	0.00	14.29	53.59	45.45	57.14	90.85	100.00	100.00	78.43	77.27	85.71
Qwen3.5-27B	29.41	18.18	42.86	47.71	40.91	71.43	92.81	95.45	100.00	70.59	81.82	71.43
Qwen3.5-9B	14.38	9.09	0.00	38.56	27.27	71.43	78.43	86.36	85.71	59.48	72.73	71.43
Qwen3.5-4B	7.19	0.00	14.29	26.14	27.27	28.57	69.93	72.73	85.71	54.25	68.18	57.14
DeepSeek-R1 Distill												
R1-0528-8B	5.88	4.55	28.57	15.69	27.27	14.29	67.97	81.82	42.86	37.25	54.55	42.86
R1-Distill-32B	6.54	4.55	0.00	15.03	31.82	42.86	64.05	68.18	85.71	41.83	59.09	28.57
R1-Distill-14B	4.58	4.55	0.00	15.69	18.18	14.29	68.63	54.55	71.43	37.25	54.55	28.57
R1-Distill-7B	7.84	4.55	0.00	13.73	9.09	14.29	49.02	50.00	42.86	30.07	45.45	42.86

Table 7: Performance on PRIM across the four dimensions of mathematical understanding, stratified by primitive family: **R** = Recast, **W** = Witness, and **A** = Argument.

B.3.2 Reasoning Length and Generation Behavior

Figure 12 compares the output-token distributions of representative models during Generation. The left panel shows the overall distribution across all rollouts, while the right panel separates rollouts by final-answer correctness, providing a view of how reasoning length varies across models and between successful and unsuccessful solutions.



(a) All rollouts (b) By correctness

Figure 12: Output-token distributions of representative models from each model family for **Generation**. Left: all rollouts. Right: rollouts separated by final-answer correctness.

B.3.3 Execution under Different Primitive Sources

Table 8 provides the full results underlying the Execution analysis in §3, extending the comparison from the representative models in Table 2 to all 12 evaluated models. The same pattern holds

consistently: conditioning on self-generated primitives provides little or no benefit and often degrades performance, whereas externally supplied teacher primitives substantially improve Execution across model families, with gold primitives yielding the largest gains. These results further support that the rescue effect is not a consequence of simply decomposing generation into a discover-then-execute pipeline; rather, successful execution depends on supplying the relevant mathematical structure that the model often fails to identify on its own.

Model	Generation	Execution		
		Self-Generated Primitive	Teacher Primitive	Gold Primitive
OpenAI				
gpt-5.4	67.03	74.18 (+7.15)	—†	86.81 (+19.78)
gpt-5.4-mini	50.55	48.90 (-1.65)	63.74 (+13.19)	71.98 (+21.43)
gpt-5.4-nano	44.51	46.15 (+1.64)	60.99 (+16.48)	68.68 (+24.17)
gpt-oss-20b	43.41	38.46 (-4.95)	58.24 (+14.83)	60.99 (+17.58)
Qwen				
Qwen3.6-27B	52.75	41.21 (-11.54)	64.84 (+12.09)	78.57 (+25.82)
Qwen3.5-27B	47.80	36.81 (-10.99)	58.24 (+10.44)	71.98 (+24.18)
Qwen3.5-9B	38.46	23.08 (-15.38)	52.75 (+14.29)	61.54 (+23.08)
Qwen3.5-4B	26.37	17.03 (-9.34)	42.31 (+15.94)	56.04 (+29.67)
DeepSeek-R1 Distill				
R1-0528-Qwen3-8B	17.03	10.99 (-6.04)	31.32 (+14.29)	39.56 (+22.53)
R1-Distill-Qwen-32B	18.13	17.58 (-0.55)	37.91 (+19.78)	43.41 (+25.28)
R1-Distill-Qwen-14B	15.93	14.84 (-1.09)	31.87 (+15.94)	39.01 (+23.08)
R1-Distill-Qwen-7B	13.19	10.44 (-2.75)	19.78 (+6.59)	32.42 (+19.23)

Table 8: Generation and Execution performance on PRIM under different settings. Execution is evaluated with self-generated, teacher-provided, and gold primitives. Percentage-point changes are reported relative to Generation. †The teacher-primitive result for gpt-5.4 is omitted because gpt-5.4 serves as the teacher model.

Model	n_{fail}	Composition of Generation Failures (%)			
		D^-E^+ Discovery-limited	D^-E^- Capability-limited	D^+E^- Execution-limited	D^+E^+ Utilization-limited
OpenAI					
gpt-5.4	60	15.0	10.0	21.7	53.3
gpt-5.4-mini	90	28.9	21.1	26.7	23.3
gpt-5.4-nano	101	38.6	31.7	18.8	10.9
gpt-oss-20b	103	35.0	45.6	9.7	9.7
Qwen					
Qwen3.6-27B	86	48.8	33.7	7.0	10.5
Qwen3.5-27B	95	42.1	30.5	15.8	11.6
Qwen3.5-9B	112	44.6	42.9	6.2	6.2
Qwen3.5-4B	134	42.5	52.2	3.7	1.5
DeepSeek-R1 Distill					
R1-0528-Qwen3-8B	151	28.5	65.6	2.6	3.3
R1-Distill-Qwen-32B	149	33.6	61.7	2.0	2.7
R1-Distill-Qwen-14B	153	30.7	68.0	1.3	0.0
R1-Distill-Qwen-7B	158	26.6	68.4	3.2	1.9

Table 9: Composition of direct Generation failures on PRIM. Each failed problem is categorized by whether the model discovers the gold primitive from the question (D; $\tau = 0.8$) and whether it can execute the primitive when provided (E).

B.3.4 Failure Decomposition and Rescue Analysis

Table 9 provides the full model-wise breakdown underlying Figure 4. Across model families, Generation failures are concentrated in the D^- regimes. This decomposition further clarifies the role of primitive discovery: even when overall solving ability is relatively strong, a substantial portion of failures arises because the model cannot independently identify the relevant mathematical structure, despite being able to execute it once supplied.

We further examine how Digestion relates to a model's ability to make use of externally supplied primitive guidance. Table 10 partitions initially failed problems according to whether the model can correctly recover the primitive from the reference solution, and then measures the rescue rate after supplying the teacher primitive. For most models, digestible failures are substantially more likely to be rescued than undigestible ones, suggesting that the ability to recover mathematical structure from a completed solution is closely related to the ability to incorporate that structure into downstream reasoning. This relationship is particularly clear for the Qwen3.5 models, where digestible problems exhibit rescue-rate advantages of 17.6–28.6 percentage points. The pattern is weaker for the DeepSeek-R1 distill models and is noisy for models with very few undigestible examples, indicating that Digestion and successful execution are related but distinct capabilities.

Model	Digestion	Rescue Rate		Gap
		Digestible	Undigestible	
OpenAI				
gpt-5.4-mini†	97.8	42.5	33.3	9.2
gpt-5.4-nano†	97.3	40.8	0.0	40.8
gpt-oss-20b†	91.2	40.4	11.1	29.3
Qwen				
Qwen3.6-27B†	92.3	36.4	66.7	−30.3
Qwen3.5-27B	93.4	37.6	10.0	27.6
Qwen3.5-9B	79.7	41.6	13.0	28.6
Qwen3.5-4B	70.9	35.1	17.5	17.6
DeepSeek-R1 Distill				
R1-0528-8B	68.7	23.3	16.7	6.6
R1-Distill-32B	65.4	32.6	26.3	6.3
R1-Distill-14B	67.0	24.8	23.1	1.7
R1-Distill-7B	48.9	19.7	13.4	6.3

Table 10: Relationship between Digestion and rescue under teacher-primitive guidance. Problems are partitioned by whether the model correctly recovers the primitive from the reference solution (*digestible*) or not (*undigestible*); rescue rate is the fraction of initially incorrect problems solved after providing the teacher primitive. Gap denotes the digestible–undigestible difference in rescue rate. †Undigestible estimates are based on fewer than 20 problems and are therefore high variance.

B.3.5 Repairability across Failure Modes

To complement the main-text analysis of discovery-limited and capability-limited failures, Table 11 reports repair rates across all four initial failure quadrants of the base model. The distinction between D^-E^+ and D^-E^- remains particularly consistent: across SFT, OPSD, and ABSORB, discovery-limited failures are repaired substantially more often than capability-limited failures. The remaining quadrants exhibit different repair profiles, with utilization-limited (D^+E^+) cases showing relatively high repair rates and execution-limited (D^+E^-) cases varying more strongly across post-training methods. These results provide a fuller view of how the initial capability decomposition relates to subsequent repairability, while reinforcing the main-text observation that failures with latent execution capacity are especially amenable to post-training.

Base failure type	SFT	OPSD	ABSORB
D^-E^+: discovery-limited	**20.4%**	**21.1%**	**23.8%**
D^-E^-: capability-limited	6.8%	8.2%	10.2%
D^+E^-: execution-limited	11.1%	0.0%	18.5%
D^+E^+: utilization-limited	35.0%	30.0%	45.0%

Table 11: Post-training repair rates across initial PRIM failure quadrants.

C POST-TRAINING DATASET DETAILS

Table 12 presents the detailed composition of the 709 Mathematics Ph.D. qualifying examination problems used for our post-training experiments, categorized by source university and mathematical domain. The problems span 1991–2026 and are strictly parsed into LaTeX. We select qualifying-examination problems because their difficulty and open-ended reasoning demands closely match those of PRIM, requiring substantial mathematical maturity and structural reasoning rather than short-form answer prediction. Accordingly, the dataset consists entirely of open-ended proof problems without algorithmically verifiable short-answer labels. Each problem is paired with a human-authored reference proof (median length: 118 words) and a one-sentence primitive extracted from that proof by GPT-5.4-High using the curation prompt of Figure 7.

Source University (1991–2026)	**Problem Count**
University of Oregon	332
UW Madison	131
Harvard University	126
UC Berkeley	120
Mathematical Domain	**Problem Count**
Algebra	318
Analysis	206
Topology	97
Geometry	82
Number Theory	5
Logic	1

Table 12: Detailed composition of the training corpus.

D POST-TRAINING SETUP

All ABSORB and OPSD runs use the same training recipe and differ in the privileged context provided to the teacher: ABSORB uses the mathematical primitive, whereas OPSD uses the reference solution. Training uses the 709 qualifying-examination problems, each paired with a one-sentence primitive. For each problem, the student generates one on-policy rollout from the problem alone using temperature 1.0, top-p 0.95, top-k 20, a maximum of 4096 completion tokens, and a total context length of 6144. The teacher uses the same base model with the adapter disabled and is conditioned on the corresponding privileged context.

For ABSORB, we optimize $\mathcal{L}_{\text{ABSORB}}$ with top-K support $K = 128$ and clamp threshold $\tau = 0.06$, applied to all generated completion tokens, including truncated rollouts. We train LoRA adapters with rank 64 and $\alpha = 128$ on all attention and MLP projections, including the linear-attention projections in Qwen3.5. Optimization uses fused AdamW without weight decay, a learning rate of 5×10^{-6} with linear scheduling and no warmup, gradient clipping at 0.1, and an effective batch size of 32 with per-device batch size 1 and gradient accumulation. Training uses bf16, gradient checkpointing, and seed 42. Each model is trained for one epoch, corresponding to 22–23 optimizer steps. On-policy rollouts are generated with a co-located vLLM engine synchronized with the student after every optimizer step.

E POST-TRAINING ABLATION STUDIES

To validate the design choices of ABSORB, we conduct ablation studies on the Qwen3.5-9B baseline using PRIM ($n = 182$). We organize the analysis around three questions: what information should be provided to the teacher as privilege, how that privileged guidance should be transferred to the student, and whether the primitive should instead be learned as an explicit prediction target. For all comparisons, we report Generation and Discovery together with absolute changes relative to the base model.

Variant	Generation	Discovery
Qwen3.5-9B (base)	38.46	13.19
Solution privilege	33.52 (-4.95)	10.99 (-2.20)
Primitive privilege (ABSORB)	**43.96** (+5.49)	**14.29** (+1.10)

Table 13: Ablation on the source of privileged information using Qwen3.5-9B. The training objective is held fixed while the teacher is conditioned on either the full reference solution or the mathematical primitive.

Teacher Conditioning Table 13 isolates the effect of the teacher's privileged input. In both variants, the student follows the same on-policy solution rollout and is trained with the same ABSORB objective; the only difference is whether the teacher conditions on the full reference solution or on the mathematical primitive. Under this controlled comparison, solution conditioning reduces Generation from 38.46 to 33.52, whereas primitive conditioning improves it to 43.96. Discovery follows the same trend. This shows that the gain does not arise from privileged conditioning alone: the primitive provides a more targeted supervisory signal than the full solution.

Transfer Mechanism Table 14 fixes the mathematical primitive as the teacher's privileged input and keeps the student rollout in solution space, while varying how the primitive-conditioned teacher signal is transferred. The *rKL, no cap* variant applies reverse KL without restricting teacher-induced changes, providing the unconstrained baseline. The *fKL, capped* variant instead uses forward KL on the teacher's top-K support while retaining the same cap, testing whether the benefit comes simply from limiting large contributions rather than from the asymmetric reverse-KL formulation. Both variants underperform the base model, with Generation dropping to 31.87 and 36.81, respectively.

Transfer Mechanism	Generation	Discovery
Qwen3.5-9B (base)	38.46	13.19
rKL, no cap	31.87 (-6.59)	10.44 (-2.75)
fKL, capped	36.81 (-1.65)	12.09 (-1.10)
Divergence control	34.62 (-3.85)	12.09 (-1.10)
Entropy control	31.87 (-6.59)	14.29 (+1.10)
Bounded override (ABSORB)	**43.96** (+5.49)	**14.29** (+1.10)

Table 14: Ablation on mechanisms for transferring primitive-guided teacher supervision using Qwen3.5-9B. All variants use the mathematical primitive as privileged information.

Divergence control tests whether harmful transfer can be identified purely by the magnitude of teacher–student disagreement. Let

$$D_t = D_{\mathrm{KL}}(\bar{\pi}_S(\cdot \mid \hat{y}_{<t}, x) \,\|\, \bar{\pi}_T(\cdot \mid \hat{y}_{<t}, x, p)),$$

where both distributions are renormalized. Since the divergence distribution is highly concentrated in a small tail, we mask tokens whose divergence exceeds the offline 95th-percentile threshold:

$$m_t^{\mathrm{div}} = \mathbf{1}[D_t \le \tau_D], \qquad \tau_D = 0.05.$$

This removes only about 5% of tokens but approximately 86% of the total divergence mass, partially recovering Generation from 31.87 to 34.62.

Entropy control tests a finer hypothesis: whether harmful overrides and useful primitive guidance can be separated within this high-divergence tail using the student's confidence. We define the student's full-vocabulary entropy as

$$H_t = -\sum_v \pi_S(v \mid \hat{y}_{<t}, x) \log \pi_S(v \mid \hat{y}_{<t}, x),$$

and mask only high-divergence, low-entropy positions:

$$m_t^{\text{ent}} = \mathbf{1}[D_t \le \tau_D \ \vee \ H_t \ge \tau_H], \qquad \tau_H = 0.86.$$

This preserves high-divergence positions where the student is uncertain, under the hypothesis that these are more likely to contain useful primitive-specific guidance. However, despite masking fewer tokens, Generation returns to 31.87, indicating that neither divergence magnitude nor student confidence cleanly separates beneficial from harmful teacher intervention.

In contrast, ABSORB avoids making a whole-token keep-or-drop decision. Its bounded override operates within each token distribution, retaining supervision at every position while limiting how strongly individual teacher preferences can suppress student-supported choices. This is the only transfer mechanism in the comparison that improves over the base model, reaching 43.96 Generation.

Rollout Target Table 15 isolates whether the mathematical primitive should serve as privileged guidance or as the student's rollout target. In the ABSORB setting, the student continues to generate solutions while the primitive is available only to the teacher. We compare this against variants in which the student instead generates the primitive itself under uncapped reverse KL, capped forward KL, or the same capped reverse-KL objective used by ABSORB. Primitive-target rollouts remain weak across all three objectives. Even under the ABSORB objective, changing only the rollout target from solution to primitive yields 39.01 Generation, compared with 43.96 when the primitive instead conditions the teacher while the student generates solutions. This indicates that the primitive is more effective as privileged guidance that shapes solution reasoning than as the object the student is directly optimized to produce.

Rollout Target / Objective	Generation	Discovery
Qwen3.5-9B (base)	38.46	13.19
ABSORB	**43.96** (+5.49)	**14.29** (+1.10)
rKL, no cap	32.97 (-5.49)	13.19 (+0.00)
fKL, capped	33.52 (-4.95)	9.34 (-3.85)
rKL, capped	39.01 (+0.55)	12.09 (-1.10)

Table 15: Ablation on the student rollout target using Qwen3.5-9B. The ABSORB baseline keeps the student in solution-generation space and uses the primitive only as privileged teacher information; the remaining variants instead train the student to generate the primitive itself under different distillation objectives.

Alternative Training Objectives Table 16 compares ABSORB against alternative post-training objectives on the same Qwen3.5-9B backbone. We separate the comparison into teacher-guided distillation objectives and direct supervised fine-tuning objectives.

Among the distillation baselines, *TOP-D* (Xie et al., 2026) is a policy-gradient on-policy distillation method based on a bounded proximal reward. We adapt it to our setting using the same 709-problem training corpus, primitive privilege, and self-teacher as ABSORB, with $G = 4$ student rollouts per prompt, proximal reward

$$\tilde{r}_t = \log(\alpha \rho_t + 1 - \alpha), \qquad \alpha = 0.1,$$

followed by token-level length-normalized future returns and group-normalized advantages. Despite operating with the same privileged signal, TOP-D reduces Generation to 31.87 and Discovery to 9.34, indicating that the primitive alone is insufficient without an appropriate transfer mechanism.

Dual-channel is an alternative design motivated by our analysis of rescued trajectories, which suggested that primitive guidance primarily acts through early route selection and later adjudication. The route channel applies reverse KL only to the first 512 completion tokens. The adjudication channel operates at student-generated self-doubt forks (e.g., "Wait", "But", "However", "Actually") and asks the student to sample candidate continuations. The primitive-conditioned teacher provides only a preference over these student-generated candidates rather than generating tokens itself. Given teacher and frozen-base scores z_T and z_0, we form the base-corrected teacher preference

$$q_T = \mathrm{softmax}\left(\frac{z_T - z_0}{0.2}\right),$$

and optimize

$$\mathcal{L}_{\mathrm{adj}} = D_{\mathrm{KL}}(q_T \parallel q_S),$$

with weight 0.3 in addition to the early-route reverse-KL term. Because this variant uses a larger 1,692-problem pool, full-length primitive annotations, and 8k-token completions, it should be interpreted as a reference alternative rather than a strict single-variable ablation. It nevertheless substantially underperforms both the base model and ABSORB.

We separately compare direct supervised objectives. *Solution SFT* fine-tunes the model on the human-authored reference solutions, while *primitive SFT* directly trains the model to generate the mathematical primitive. Solution SFT lowers Generation to 32.42, and primitive SFT performs worst overall, reducing Generation and Discovery to 28.02 and 2.75, respectively. Together, these results show that neither alternative teacher-guided distillation nor direct supervision on solutions or primitives substitutes for the combination used by ABSORB: primitive-conditioned teacher guidance transferred through bounded override while the student remains in solution-generation space.

Training Objective	**Generation**	**Discovery**
Qwen3.5-9B (base)	38.46	13.19
ABSORB	**43.96** (+5.49)	**14.29** (+1.10)
Teacher-guided distillation		
TOP-D (Xie et al., 2026)	31.87 (-6.59)	9.34 (-3.85)
Dual-channel†	29.67 (-8.79)	10.44 (-2.75)
Direct supervised fine-tuning		
Solution SFT	32.42 (-6.04)	12.64 (-0.55)
Primitive SFT	28.02 (-10.44)	2.75 (-10.44)

Table 16: Comparison of alternative post-training objectives using Qwen3.5-9B. Teacher-guided distillation baselines include TOP-D (Xie et al., 2026) and our dual-channel alternative, while the SFT baselines directly supervise either the reference solution or the mathematical primitive. †Dual-channel uses a larger 1,692-problem training pool, full-length primitive annotations, and 8k-token completions, and is therefore included as a reference alternative rather than a strict single-variable ablation.