# RETHINKING GROUP ACCURACY: LATENT-RELIABILITY INFERENCE FOR PROMPT SOLVABILITY IN RLVR

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

## ABSTRACT

In reinforcement learning with verifiable rewards (RLVR), group relative policy optimization (GRPO) constructs relative advantages by centering response rewards with the empirical group-average reward, a finite-sample estimate of prompt solvability under the current LLM. However, empirical group accuracy depends only on the success count and treats equally rewarded responses as equally informative. Under heterogeneous response reliability, this can misestimate prompt solvability and miscalibrate relative advantages. Consistent with this concern, we empirically observe substantial reliability variation among prompts with identical group accuracy, revealing that the same success count can mask markedly different response-level reliability patterns. In this work, we formulate prompt-solvability estimation as a latent-variable inference problem and introduce Reliability-Aware Latent Solvability Inference (RLSI). Our approach introduces a latent response-level success state and relates each verifier outcome to this state through its response-specific reliability. Under this latent-variable model, we derive an efficient expectation–maximization (EM) procedure to estimate prompt solvability and use the resulting estimate for advantage construction. Theoretically, we show why reward-only aggregation can fail to recover prompt solvability, while response-level reward–reliability pairings retain information relevant to latent-state inference; we further characterize the resulting reliability-dependent corrections and recover the GRPO group mean in the perfect-reliability limit. Across six mathematical reasoning benchmarks and two model scales, RLSI achieves the strongest average performance among the evaluated methods, outperforming GRPO by 5.2 and 4.8 points on Qwen3-1.7B-Base and Qwen3-4B, respectively. Additional comparisons and ablations further support the effectiveness of latent solvability inference and the value of response-level reliability heterogeneity.

## 1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) has become a central paradigm for improving the reasoning capabilities of large language models (LLMs), particularly in domains such as mathematics and coding where the correctness of a final answer can be automatically verified (Shao et al., 2024; Guo et al., 2025). Group Relative Policy Optimization (GRPO) (Shao et al., 2024) and its variants (Mroueh, 2025; Yu et al., 2026; Liu et al., 2025) sample multiple responses to each prompt and construct relative advantages by subtracting their empirical group accuracy from each binary verifier reward. This makes the estimate of prompt-level performance central to the learning signal: changing the shared baseline changes the advantages assigned to every response in the group.

However, empirical group accuracy depends only on the success count: responses with the same binary verifier reward contribute equally to the estimate, regardless of their response-specific reliability. When reliability is heterogeneous, equally rewarded responses need not provide equally strong evidence about latent success. As a result, two response groups with the same success count can imply different prompt solvability under a reliability-aware observation model, while empirical group accuracy assigns them the same baseline. This can misestimate prompt solvability and miscalibrate the relative advantages constructed from that baseline, especially when only a small group of responses is available.



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.

Recent methods have sought to enrich the learning signals obtained from binary verifier rewards. ConfClip (Zhang et al., 2026) uses model confidence to reweight response rewards, whereas Self-Aligned Reward (Han et al., 2026) adds a model-derived score to encourage concise and prompt-relevant reasoning. At the prompt level, CoDaPO (Zhou et al., 2026) combines rollout confidence and empirical difficulty to reweight policy updates and prioritize learnable prompts. These approaches use model-derived scores to transform response-level rewards or adjust the contribution of different prompts. However, they do not formulate the shared baseline in group-relative optimization as an explicit prompt-solvability estimation problem. Consequently, it remains unclear how verifier outcomes and heterogeneous response reliability should be jointly aggregated into the prompt-level baseline that determines relative advantages.

This estimation issue is reflected in Figure 1. Among prompts with the same success count, we observe substantial variation in response reliability. Although empirical group accuracy assigns these prompts the same baseline, their reward–reliability pairings can provide different evidence about latent success under a reliability-aware observation model. This motivates retaining the response-level pairings when estimating the shared prompt-level baseline.

Motivated by this observation, we introduce *Reliability-Aware Latent Solvability Inference* (RLSI), which recasts prompt-solvability estimation as a latent-variable inference problem rather than treating verifier outcomes as direct observations of solvability. For each response, we introduce a binary latent success state, and define prompt solvability as the probability of latent success for a given prompt under the current policy. Rather than identifying latent success with observed final-answer correctness, we specify an observation model in which a response's reliability determines how its verifier outcome informs the latent state. Under this latent-variable model, we derive an efficient expectation–maximization (EM) procedure to estimate the prompt-solvability. In the E-step, we compute each response's posterior probability of latent success using its verifier reward, reliability score, and the current prompt-level solvability estimate. In the M-step, we average these posterior probabilities to update the solvability estimate, maximizing the expected complete-data log-likelihood with respect to that parameter. The updated estimate replaces empirical group accuracy in reward subtraction and sets the advantage normalization scale within the GRPO optimization framework.

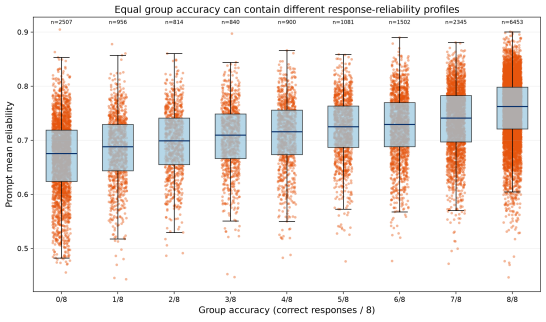


Figure 1: Mean response reliability across prompts grouped by the number of correct responses among eight samples. Each point denotes one prompt.

The central advantage of RLSI is that it gives response-specific reliability an explicit role in estimating the prompt-level latent success probability. Introducing the latent response state separates underlying solving success from the observed verifier outcome, allowing reliability to determine how strongly each outcome supports latent success rather than using it as a heuristic reward weight. This latent formulation makes the uncertainty behind binary verifier outcomes explicit: the E-step represents each unobserved state through its posterior probability, and the M-step aggregates these probabilities through a closed-form solvability update. Consequently, RLSI can distinguish response groups with identical success counts when their reward–reliability pairings provide different evidence, yielding different prompt-level baselines and normalization scales for relative advantage construction. From a statistical perspective, the posterior representation retains the response-level structure needed for latent-state inference before aggregation, which our theoretical analysis shows can be lost when verifier rewards are reduced directly to their group average. Our main contributions are summarized as follows:

- We formulate prompt-solvability estimation in group-relative optimization as a latent-variable inference problem, separating latent response success from observed verifier outcomes and using response-specific reliability to model their relationship. Building on this formulation, we derive an efficient EM estimator of prompt solvability and use the resulting estimate for GRPO-style advantage construction.
- Under the proposed observation model, we show that reward-only aggregation generally does not recover prompt solvability. We further establish that response-level reward–

reliability pairings retain information needed for latent-state inference, characterize their reliability-dependent effect on solvability estimation, and recover the GRPO group mean in the perfect-reliability limit.

- Empirically, we evaluate RLSI across six mathematical reasoning benchmarks and two Qwen3 model scales, with consistent improvements over strong RLVR baselines. Beyond overall performance, targeted comparisons and ablations further support the distinct roles of latent solvability inference and response-level heterogeneity in the reliability signal.

## 2 PRELIMINARIES

**RLVR and response reliability.** We consider reinforcement learning with verifiable rewards (RLVR) over a training prompt set $\mathcal{X} = \{x_i\}_{i=1}^{N}$. At RL iteration $t$, the behavior policy $\pi_{\theta_{t-1}}$ independently generates $G$ responses for each sampled prompt, where $y_{ij}^{(t)} \sim \pi_{\theta_{t-1}}(\cdot \mid x_i)$ for $j = 1, \ldots, G$. Each response is evaluated by a verifier $\mathcal{V}$, producing the binary outcome reward $R_{ij}^{(t)} = \mathcal{V}(x_i, y_{ij}^{(t)}) \in \{0, 1\}$. In addition to the verifier reward, we associate each response with a response-specific reliability score $c_{ij}^{(t)} \in (0, 1)$. This score reflects the reliability of the evidence provided by response $y_{ij}^{(t)}$ when inferring prompt solvability. The framework does not prescribe a particular function for computing $c_{ij}^{(t)}$. It requires the score to lie in $(0, 1)$, where $c_{ij}^{(t)} = 1/2$ represents uninformative evidence, values above $1/2$ support agreement between the verifier outcome and the latent state, and values below $1/2$ support disagreement.

**Group Relative Policy Optimization.** Group Relative Policy Optimization (GRPO) constructs relative learning signals from responses generated for the same prompt. For prompt $x_i$, its empirical group reward is $\bar{R}_i^{(t)} = G^{-1} \sum_{j=1}^{G} R_{ij}^{(t)}$, and its group standard deviation is $s_i^{(t)} = \left[G^{-1} \sum_{j=1}^{G} (R_{ij}^{(t)} - \bar{R}_i^{(t)})^2 + \epsilon\right]^{1/2}$. The corresponding advantage is $A_{ij}^{\mathrm{GRPO},(t)} = (R_{ij}^{(t)} - \bar{R}_i^{(t)})/s_i^{(t)}$. GRPO then optimizes the standard clipped surrogate objective with reference-policy KL regularization (Shao et al., 2024). Its group-dependent learning signal is determined by $A_{ij}^{\mathrm{GRPO},(t)}$. Because both $\bar{R}_i^{(t)}$ and $s_i^{(t)}$ depend only on the binary reward vector, they cannot distinguish groups with the same success count but different reliability patterns. RLSI retains the GRPO policy-optimization objective while replacing these reward-only statistics with a reliability-aware estimate of prompt solvability.

## 3 METHOD

The central challenge identified in Section 1 is that empirical group accuracy reduces a response group to its success count. Consequently, responses receiving the same verifier reward contribute equally to the shared baseline, even when their outcomes have different reliability. Under heterogeneous response reliability, this reward-only aggregation can provide an inaccurate estimate of prompt solvability and consequently distort the relative advantages constructed from that estimate.

We address the limitations of empirical group accuracy by formulating prompt-solvability estimation as a latent-variable inference problem. Specifically, we introduce a latent response state that separates underlying success from the observed verifier outcome. Rather than identifying the verifier outcome directly with the latent state, we model their relationship through response-specific reliability. In our setting, prompt solvability is the estimand of interest, defined as the probability of latent success under the current policy. All responses to the same prompt share this prompt-level solvability, while their individual reliability scores determine how their observed verifier outcomes inform the corresponding latent states. This allows responses with the same verifier outcome to contribute differently to the resulting solvability estimate.

Because the response states are unobserved, estimating prompt solvability is an incomplete-data maximum-likelihood problem. We therefore derive an expectation–maximization procedure that alternates between computing response-level posterior probabilities of latent success and updating the shared prompt-solvability estimate. The resulting estimate replaces empirical group accuracy in

reward subtraction and determines the corresponding normalization scale for GRPO-style advantage construction.

### 3.1 Latent-Variable Model for Prompt Solvability

We first formalize the latent-variable model that connects response-specific reliability to prompt solvability. We consider a fixed policy and omit the RL iteration index $t$. For each prompt $x_i$, we observe verifier rewards $R_{ij}$ and response-specific reliability scores $c_{ij}$ for $j = 1, \ldots, G$. For each response, we introduce a binary latent state $Z_{ij} \in \{0, 1\}$, where $Z_{ij} = 1$ denotes latent success.

All responses sampled for the same prompt share a common success probability under the fixed policy. Because the latent state represents a binary success event and the parameter of interest is its probability, we model it using a Bernoulli distribution:

$$Z_{ij} \mid q_i \sim \text{Bernoulli}(q_i), \qquad q_i = P(Z_{ij} = 1 \mid x_i, \pi_\theta). \tag{3.1}$$

We refer to $q_i$ as prompt solvability under the proposed model. It is shared by all responses to prompt $x_i$ and therefore provides the prompt-level quantity that will later serve as the common baseline. The verifier reward $R_{ij}$ is the observed correctness outcome, whereas $Z_{ij}$ is the latent response state through which this outcome contributes to the estimation of $q_i$. Together with the reliability-dependent observation model introduced below, this distinction allows responses with the same verifier reward to exert different influence on the resulting solvability estimate.

The latent state $Z_{ij}$ is not directly observed; training instead provides the verifier reward $R_{ij}$. We therefore construct a probabilistic observation model that describes how the reliability score $c_{ij}$ affects the relationship between the observed reward and the latent state. Specifically, we model the verifier reward as matching the latent state with probability $c_{ij}$:

$$P(R_{ij} = Z_{ij} \mid Z_{ij}, c_{ij}) = c_{ij}, \qquad P(R_{ij} \neq Z_{ij} \mid Z_{ij}, c_{ij}) = 1 - c_{ij}. \tag{3.2}$$

Under this model, the reliability score determines how strongly the observed verifier reward supports either value of the latent state. Thus, responses with the same verifier reward can have different effects on solvability estimation when their reliability scores differ.

Thus, all responses to the same prompt share the same latent-success prior; reliability enters through the observation model rather than through a response-specific prior. Conditional on $q_i$ and the reliability vector $\mathbf{c}_i$, the latent states are independent, and each reward depends only on its corresponding latent state and reliability score. These assumptions yield the factorized conditional model.

$$p(\mathbf{Z}_i, \mathbf{R}_i \mid q_i, \mathbf{c}_i) = \prod_{j=1}^{G} p(Z_{ij} \mid q_i) p(R_{ij} \mid Z_{ij}, c_{ij}). \tag{3.3}$$

Marginalizing over the latent state $Z_{ij}$ gives the conditional likelihood of each observed reward:

$$\begin{aligned} P(R_{ij} \mid q_i, c_{ij}) &= \sum_{z \in \{0,1\}} P(Z_{ij} = z \mid q_i) P(R_{ij} \mid Z_{ij} = z, c_{ij}) \\ &= q_i c_{ij}^{R_{ij}} (1 - c_{ij})^{1 - R_{ij}} + (1 - q_i)(1 - c_{ij})^{R_{ij}} c_{ij}^{1 - R_{ij}}. \end{aligned} \tag{3.4}$$

This marginalization clarifies the distinction between latent prompt solvability and observed verifier accuracy. For response $j$, the probability of receiving a positive verifier reward is $P(R_{ij} = 1 \mid q_i, c_{ij}) = q_i c_{ij} + (1 - q_i)(1 - c_{ij})$, which generally differs from $q_i$. Thus, observed verifier accuracy generally differs from latent prompt solvability. Empirical group accuracy aggregates only the observed outcomes, whereas the proposed model also retains how those outcomes are paired with response-specific reliability. Table 1 illustrates that this pairing can change the inferred solvability even when the reliability vector and success count are fixed.

Holding both the reliability vector and empirical group accuracy fixed, changing only which reliability scores are paired with positive rewards produces different solvability estimates after five fixed-group EM iterations from the same initialization. Thus, empirical group accuracy discards response-level pairing information that is relevant to inferring latent prompt solvability. Estimating this latent quantity from the observed reward–reliability pairs requires accounting for uncertainty in the unobserved states $\mathbf{Z}_i$. We therefore formulate solvability estimation as an incomplete-data maximum-likelihood problem and derive an EM procedure in the next subsection.

Table 1: Toy reward–reliability pairings with fixed $\mathbf{c} = (0.9, 0.8, 0.7, 0.6)$, $\bar{R} = 0.5$, and $q^{(0)} = 0.5$. The reported estimates follow five fixed-group EM iterations.

| Reliabilities at $R = 1$ | $\mathbf{R}$ | $\bar{R}$ | $q^{(5)}$ |
|---|---|---|---|
| $\{0.9, 0.8\}$ | $(1, 1, 0, 0)$ | 0.5 | 0.784 |
| $\{0.9, 0.6\}$ | $(1, 0, 0, 1)$ | 0.5 | 0.500 |
| $\{0.7, 0.6\}$ | $(0, 0, 1, 1)$ | 0.5 | 0.216 |

### 3.2 Prompt-Solvability Inference via EM

The latent formulation above provides a probabilistic model for prompt solvability, but its response-level success states are not observed. If these states were available, the maximum-likelihood estimate of $q_i$ would be obtained directly from their empirical mean. With only verifier rewards and response reliability scores observed, however, this latent success count cannot be computed, and maximizing the observed-data likelihood does not yield the same direct update. We therefore develop an expectation–maximization (EM) procedure under the proposed latent solvability model. We first derive the EM updates for a fixed response group and then describe how they are used for offline initialization and online policy training.

For a fixed response group, the parameter $q_i$ is estimated by maximizing the conditional log-likelihood of the observed rewards:

$$\ell_i(q_i) = \sum_{j=1}^{G} \log P(R_{ij} \mid q_i, c_{ij}). \tag{3.5}$$

Let $k$ index EM iterations. In the E-step, using the current estimate $q_i^{(k)}$, we compute the posterior probability that each response's latent success state equals one:

$$\gamma_{ij}^{(k)} = P(Z_{ij} = 1 \mid R_{ij}, c_{ij}, q_i^{(k)}) = \frac{q_i^{(k)} c_{ij}^{R_{ij}} (1 - c_{ij})^{1 - R_{ij}}}{q_i^{(k)} c_{ij}^{R_{ij}} (1 - c_{ij})^{1 - R_{ij}} + (1 - q_i^{(k)})(1 - c_{ij})^{R_{ij}} c_{ij}^{1 - R_{ij}}}. \tag{3.6}$$

We denote this posterior map by $\Gamma(R_{ij}, c_{ij}; q_i^{(k)})$. $\gamma_{ij}^{(k)}$ is a response-level posterior probability, whereas $q_i^{(k)}$ is the shared prompt-level parameter used as its prior. Thus, responses to the same prompt can receive different posterior probabilities despite sharing the same prior. In the M-step, the part of the expected complete-data log-likelihood that depends on $q_i$ is

$$Q_i(q_i \mid q_i^{(k)}) = \sum_{j=1}^{G} \left[ \gamma_{ij}^{(k)} \log q_i + (1 - \gamma_{ij}^{(k)}) \log(1 - q_i) \right],$$

where terms independent of $q_i$ are omitted. Maximizing this expression yields $q_i^{(k+1)} = G^{-1} \sum_{j=1}^{G} \gamma_{ij}^{(k)}$. Thus, the M-step updates the prompt-level solvability estimate by averaging the posterior probabilities of latent success across the responses in the group.

Before RL training, we initialize $q_i$ using responses sampled from the reference policy $\pi_{\text{ref}}$. For each prompt $x_i$, we sample $G$ responses, compute their verifier rewards and reliability scores, initialize the offline EM estimate at $q_0$, and run $K_{\text{off}}$ EM iterations on this fixed offline group. This gives an initial estimate $q_i^{\text{init}}$ for every training prompt. At the start of RL training, we set $q_i^{(0)} = q_i^{\text{init}}$; thereafter, the superscript $t$ indexes RL iterations. The offline EM iterations do not add new samples; they only refine the latent solvability estimate under the fixed reference-policy responses.

The offline estimate is obtained under the reference policy, whereas policy training changes the response distribution. We therefore update each visited prompt using a fresh group sampled from the current behavior policy. The previous estimate serves as a warm start for the current group's likelihood. To incorporate the new observations without running multiple EM iterations at every visit, we perform one E-step followed by one M-step:

$$q_i^{(t)} = \frac{1}{G} \sum_{j=1}^{G} \Gamma(R_{ij}^{(t)}, c_{ij}^{(t)}; q_i^{(t-1)}) \tag{3.7}$$

Unlike the offline stage, the online update is not iterated to convergence on each newly sampled group. It should therefore be interpreted as a one-step tracking update initialized by the previous solvability estimate, rather than as the exact maximizer of the current group's likelihood. For prompts outside the current minibatch, we retain the previous estimate. For visited prompts, the update combines current observations with dependence on earlier estimates through the initialization. We use the updated estimate $q_i^{(t)}$ to construct the baseline and normalization scale for the policy advantages described next.

### 3.3 Reliability-Aware Advantage

After obtaining the reliability-aware estimate of prompt solvability, it remains to incorporate this estimate into the group-relative advantage. In GRPO, the empirical group-average reward $\bar{R}_i^{(t)}$ serves as a finite-sample estimate of prompt solvability and is used both to center the binary rewards and to determine their normalization scale. However, because $\bar{R}_i^{(t)}$ depends only on the number of successful responses, it does not account for heterogeneous response reliability. We therefore replace this reward-only solvability estimate with $q_i^{(t)}$, which incorporates the reward–reliability pairing of each response through the latent solvability model. For binary outcomes, the Bernoulli-form variance associated with a solvability estimate $q_i^{(t)}$ is $q_i^{(t)}(1-q_i^{(t)})$. Accordingly, we use $q_i^{(t)}$ both as the shared prompt-level baseline and in the corresponding normalization scale:

$$A_{ij}^{\mathrm{RA},(t)} = \frac{R_{ij}^{(t)} - q_i^{(t)}}{\sqrt{q_i^{(t)}(1-q_i^{(t)}) + \epsilon}}. \tag{3.8}$$

The policy is then updated with the same GRPO objective, replacing $A_{ij}^{\mathrm{GRPO},(t)}$ by $A_{ij}^{\mathrm{RA},(t)}$. All quantities used to estimate $q_i^{(t)}$ are treated as fixed statistics when optimizing the policy on the current batch.

The effect on reward subtraction can be seen from

$$R_{ij}^{(t)} - q_i^{(t)} = (R_{ij}^{(t)} - \bar{R}_i^{(t)}) + (\bar{R}_i^{(t)} - q_i^{(t)}). \tag{3.9}$$

The first term compares a response with the current group's empirical accuracy, while the second is a shared prompt-level correction determined by the solvability estimate. The correction is positive when the current group accuracy exceeds the estimated solvability and negative when it falls below it. It shifts the unnormalized learning signals for all responses in the group, while the common normalization scale controls their magnitude. The full training procedure, including offline initialization, online solvability updates, and policy optimization, is summarized in Algorithm 1.

## 4 Theoretical Analysis

In this section, we analyze how the proposed estimator retains response-level reward–reliability information, how reliability modifies latent-success posteriors, and how these corrections determine the prompt-level solvability update and resulting advantages.

We consider a fixed prompt $x_i$ and a fixed EM iteration index $k$, omitting the RL iteration index until needed. Let $\mathbf{R}_i = (R_{i1}, \ldots, R_{iG})$, $\mathbf{Z}_i = (Z_{i1}, \ldots, Z_{iG})$, $\mathbf{c}_i = (c_{i1}, \ldots, c_{iG})$, and $\boldsymbol{\gamma}_i^{(k)} = (\gamma_{i1}^{(k)}, \ldots, \gamma_{iG}^{(k)})$. We also define $S_i = \sum_{j=1}^{G} Z_{ij}$ as the number of latent successful responses in the group. Under the model in Section 3, the response-level latent states are conditionally independent given $q_i^{(k)}$, and each reward is generated from its corresponding latent state through the reliability parameter $c_{ij}$.

Before characterizing the posterior representation produced by the E-step, we first establish why directly aggregating the observed verifier rewards does not generally recover the latent prompt solvability under the proposed observation model.

**Proposition 4.1.** *Let $\widehat{q}_i^{\mathrm{GA}} = \bar{R}_i = G^{-1}\sum_{j=1}^{G} R_{ij}$ denote the group-average estimator and let $\widehat{q}_i^{\mathrm{RW}} = (\sum_{j=1}^{G} c_{ij}R_{ij})/(\sum_{j=1}^{G} c_{ij})$ denote the reliability-weighted estimator. Under the proposed*

*observation model, their conditional biases for $q_i$ are*

$$\mathbb{E}[\widehat{q}_i^{\mathrm{GA}} \mid q_i, \mathbf{c}_i] - q_i = (1 - \bar{c}_i)(1 - 2q_i),$$
$$\mathbb{E}[\widehat{q}_i^{\mathrm{RW}} \mid q_i, \mathbf{c}_i] - q_i = (1 - 2q_i)\frac{\sum_{j=1}^{G} c_{ij}(1 - c_{ij})}{\sum_{j=1}^{G} c_{ij}}, \tag{4.1}$$

*where $\bar{c}_i = G^{-1}\sum_{j=1}^{G} c_{ij}$. Hence, if $c_{ij} \in (0, 1)$, both estimators are biased whenever $q_i \neq 1/2$.*

Proposition 4.1 shows that direct weighting changes the relative contributions of the observed rewards but does not generally correct their mismatch with latent solvability. This motivates inferring the latent response states before aggregation. We next characterize the response-level posterior representation produced by the E-step. The posterior vector $\boldsymbol{\gamma}_i^{(k)}$ records the latent-success probability of each response after conditioning on its verifier reward and reliability score. The following result shows that these response-level probabilities completely determine the posterior distribution of the latent response states.

**Theorem 4.2 (Posterior Representation of Latent States).** *Fix a prompt $x_i$ and an EM iterate $q_i^{(k)} \in (0, 1)$. Under the latent solvability model, the posterior distribution of $\mathbf{Z}_i$ is determined by the response-level responsibilities $\boldsymbol{\gamma}_i^{(k)}$. Specifically, for every $\mathbf{z} \in \{0, 1\}^G$,*

$$P(\mathbf{Z}_i = \mathbf{z} \mid \mathbf{R}_i, \mathbf{c}_i, q_i^{(k)}) = \prod_{j=1}^{G} \big(\gamma_{ij}^{(k)}\big)^{z_j} \big(1 - \gamma_{ij}^{(k)}\big)^{1 - z_j}. \tag{4.2}$$

*Equivalently, $\mathbf{Z}_i \perp \mathbf{R}_i \mid \boldsymbol{\gamma}_i^{(k)}, \mathbf{c}_i, q_i^{(k)}$.*

This posterior representation preserves the response-level structure of the observations before they are aggregated into the prompt-level M-step update. We next compare the information retained by this representation with that retained by the empirical group-average reward used in GRPO.

**Corollary 4.3 (Information Retention Before Aggregation).** *Let $S_i = \sum_{j=1}^{G} Z_{ij}$ denote the latent group success count. Under the latent solvability model,*

$$I(S_i; \boldsymbol{\gamma}_i^{(k)} \mid \mathbf{c}_i, q_i^{(k)}) = I(S_i; \mathbf{R}_i \mid \mathbf{c}_i, q_i^{(k)}) \geq I(S_i; \bar{R}_i \mid \mathbf{c}_i, q_i^{(k)}). \tag{4.3}$$

*The inequality becomes strict whenever $\bar{R}_i$ fails to preserve reward information that is relevant to $S_i$ after conditioning on $\mathbf{c}_i$ and $q_i^{(k)}$.*

Corollary 4.3 shows that retaining the response-level reward–reliability pairs preserves information about the latent group state that can be lost when rewards are first reduced to their group average. The distinction arises from the pairing between rewards and reliability scores. For example, two groups can share the same reward count while pairing positive and negative rewards with different reliability scores. Their group averages are identical, but their posterior vectors and solvability estimates can differ.

Having characterized the posterior representation, we now examine how reliability changes the evidence contributed by an individual reward. The E-step admits an additive decomposition in log-odds space: the posterior log-odds equal the prompt-level prior log-odds plus a reward-dependent reliability correction.

**Theorem 4.4 (Signed Reliability Correction and GRPO Limit).** *For any $q_i^{(k)} \in (0, 1)$ and $c_{ij} \in (0, 1)$, the E-step posterior satisfies $\operatorname{logit} \gamma_{ij}^{(k)} = \operatorname{logit} q_i^{(k)} + (2R_{ij} - 1) \operatorname{logit} c_{ij}$. Consequently, $c_{ij} = 1/2$ leaves the posterior unchanged, $c_{ij} > 1/2$ moves it in the direction indicated by the observed reward, and these directions reverse when $c_{ij} < 1/2$. Moreover, as $c_{ij} \to 1$ for all $j$, we have $\gamma_{ij}^{(k)} \to R_{ij}$ and hence $q_i^{(k+1)} \to G^{-1}\sum_{j=1}^{G} R_{ij} = \bar{R}_i$, recovering the GRPO group mean.*

Theorem 4.4 characterizes the role assigned to reliability by the proposed observation model. Thus the same reliability value has opposite effects depending on the observed reward. The limiting case also clarifies the relation to GRPO: if every observed reward is treated as perfectly reliable, then the EM update collapses to the ordinary group reward average.

We next examine how these response-level changes combine in the M-step. Let

$$m_{ij}^{(k)} = P(R_{ij} \mid q_i^{(k)}, c_{ij}) = q_i^{(k)} c_{ij}^{R_{ij}} (1 - c_{ij})^{1-R_{ij}} + (1 - q_i^{(k)})(1 - c_{ij})^{R_{ij}} c_{ij}^{1-R_{ij}}$$

denote the marginal likelihood of the observed reward under the current prompt-level estimate.

**Theorem 4.5 (Prompt-Level Reliability Correction).** *Under the latent solvability model, the one-step EM update changes the prompt-level estimate according to*

$$q_i^{(k+1)} - q_i^{(k)} = \frac{q_i^{(k)}(1 - q_i^{(k)})}{G} \sum_{j=1}^{G} \frac{(2R_{ij} - 1)(2c_{ij} - 1)}{m_{ij}^{(k)}}. \tag{4.4}$$

Theorem 4.5 shows that the prompt-level solvability update is an aggregation of signed reliability-dependent corrections. A response with $R_{ij} = 1$ and $c_{ij} > 1/2$ increases the solvability estimate, while a response with $R_{ij} = 0$ and $c_{ij} > 1/2$ decreases it. Responses with $c_{ij} \approx 1/2$ produce little correction because their observed rewards provide weak evidence about the latent state. Therefore, unlike the GRPO group mean, the update is not determined only by the number of correct responses; it also depends on how the rewards are paired with the response reliability scores.

## 5 Experiments

**Training and Evaluation Setup.** We use the mathematical reasoning training dataset released by DAPO (Yu et al., 2026) and initialize our models from Qwen3-1.7B-Base and Qwen3-4B (Yang et al., 2025). During RL training, we sample four responses per prompt, corresponding to a rollout group size of $G = 4$. We evaluate all methods on six mathematical reasoning benchmarks: MATH500 (Lightman et al., 2024), Minerva Math (Lewkowycz et al., 2022), OlympiadBench (He et al., 2024), and the competition problem sets AIME24, AIME25 (MAA, 2025), and AMC23 (MAA, 2023). For MATH500, Minerva Math, and OlympiadBench, we generate one response per problem and report accuracy as Avg@1. For AIME24, AIME25, and AMC23, we generate 32 responses per problem and report the average accuracy across these responses as Avg@32. When summarizing performance across benchmarks, we report the unweighted arithmetic mean of the six benchmark scores.

Table 2: Performance comparison of Qwen3-1.7B-Base and Qwen3-4B on mathematical reasoning benchmarks. The best result in each column is shown in bold, and the second-best result is underlined.

| Method | MATH500 Avg@1 | Minerva Math Avg@1 | OlympiadBench Avg@1 | AIME24 Avg@32 | AIME25 Avg@32 | AMC23 Avg@32 | Average |
|---|---|---|---|---|---|---|---|
| Qwen3-1.7B-Base | 59.0 | 25.0 | 22.7 | 2.2 | 0.8 | 16.6 | 21.1 |
| PPO | 62.2 | 27.6 | 26.4 | 4.4 | 1.9 | 31.3 | 25.6 |
| GRPO | 64.2 | 26.8 | 27.7 | 6.0 | 4.2 | 36.3 | 27.5 |
| DAPO | <u>67.8</u> | 29.0 | <u>30.8</u> | **9.4** | <u>4.8</u> | <u>41.1</u> | <u>30.5</u> |
| RW-GRPO | 67.4 | <u>30.1</u> | 29.5 | 6.6 | 4.3 | 35.8 | 29.0 |
| RLSI | **69.8** | **32.0** | **33.6** | <u>9.0</u> | **7.8** | **43.9** | **32.7** |
| Qwen3-4B | 72.4 | 35.3 | 37.6 | 17.1 | 16.4 | 53.3 | 38.7 |
| PPO | 80.4 | 40.4 | 45.8 | 16.5 | 18.6 | 61.1 | 43.8 |
| GRPO | <u>87.0</u> | 43.8 | 54.1 | <u>31.7</u> | 23.1 | 67.0 | 51.1 |
| DAPO | 85.2 | <u>45.6</u> | <u>56.0</u> | 30.3 | <u>28.2</u> | <u>70.5</u> | **52.6** |
| RW-GRPO | 84.4 | 41.5 | 49.6 | 25.4 | 24.6 | 67.6 | 48.9 |
| RLSI | **88.2** | **46.3** | **58.8** | **32.9** | **29.4** | **79.5** | **55.9** |

**Reliability Function.** RLSI is compatible with different functions for constructing response-specific reliability. In our experiments, we use the length-normalized sequence probability as a model-derived proxy for response reliability: $c_{ij}^{(t)} := \exp\left\{|y_{ij}^{(t)}|^{-1} \sum_{k=1}^{|y_{ij}^{(t)}|} \log \pi_{\theta_{t-1}}\left(y_{ij,k}^{(t)} \mid x_i, y_{ij,<k}^{(t)}\right)\right\}$. This score is higher when the sampling policy assigns higher probability to the generated response. Averaging token log-probabilities before exponentiation reduces the direct dependence on response length, making scores more comparable across responses. For offline initializa-

tion, the score is computed under $\pi_{\text{ref}}$, whereas during RL training it is computed under the current behavior policy; in both cases, it is treated as a fixed statistic during solvability estimation.

**Baselines.** We compare RLSI with PPO (Schulman et al., 2017), GRPO (Shao et al., 2024), and DAPO (Yu et al., 2026). We additionally construct Reliability-Weighted GRPO (RW-GRPO) as a matched control to test whether the benefit of response-specific reliability can be explained by direct weighted aggregation alone. RW-GRPO uses the same response-specific reliability score as RLSI, but aggregates verifier outcomes directly rather than through the latent-variable model. For each prompt, RW-GRPO computes $q_i^{(t)} = \frac{\sum_{j=1}^{G} c_{ij}^{(t)} R_{ij}^{(t)}}{\sum_{j=1}^{G} c_{ij}^{(t)}}$, where $c_{ij}^{(t)}$ is the same response-specific reliability score defined above. It then substitutes $q_i^{(t)}$ into the advantage form in Equation 3.8 and retains the same GRPO policy-optimization objective.

**Main Results.** Table 2 shows that RLSI achieves the highest average performance at both model scales. On Qwen3-1.7B, it obtains an average score of 32.7, exceeding GRPO and DAPO by 5.2 and 2.2 points, respectively, and ranks first on five of six benchmarks. On Qwen3-4B, RLSI averages 55.9, outperforming GRPO and DAPO by 4.8 and 3.3 points, and ranks first on all six benchmarks. RLSI also outperforms RW-GRPO on every benchmark at both scales, by 3.7 and 7.0 points on average. Since RW-GRPO uses the same response-specific reliability signal but aggregates verifier outcomes directly, this comparison suggests that the gains from RLSI cannot be explained by introducing reliability into the baseline through direct weighted aggregation alone. Instead, the results support the benefit of incorporating response-level reliability through the latent-variable solvability model. Relative to GRPO, the largest gains occur on OlympiadBench, AIME25, and AMC23, indicating that the overall improvement is particularly pronounced on the more challenging competition-level benchmarks. Taken together, the results across both model scales and different group-based optimization baselines support the effectiveness of incorporating response-specific reliability into prompt-solvability estimation.

Table 3: Effect of response-level reliability heterogeneity on Qwen3-4B.

| Method | MATH500 Avg@1 | Minerva Math Avg@1 | OlympiadBench Avg@1 | AIME24 Avg@32 | AIME25 Avg@32 | AMC23 Avg@32 | Average |
|---|---|---|---|---|---|---|---|
| RLSI with Homogeneous Reliability | **89.4** | 43.0 | 57.2 | 31.0 | 25.0 | 73.3 | 53.2 |
| RLSI | 88.2 | **46.3** | **58.8** | **32.9** | **29.4** | **79.5** | 55.9 |

**Ablation Study.** To assess the contribution of response-level reliability heterogeneity, we construct a homogeneous-reliability variant. For each response group, we compute the mean reliability score $\bar{c}_i = G^{-1} \sum_{j=1}^{G} c_{ij}$ and replace each $c_{ij}$ with $\bar{c}_i$ when computing the E-step posteriors. The verifier rewards remain unchanged. This variant retains the group's mean reliability while removing reliability differences across its responses. Table 3 shows that, on Qwen3-4B, RLSI achieves an average benchmark score of 55.9, compared with 53.2 for the homogeneous-reliability variant, a gain of 2.7 percentage points. These results suggest that retaining response-specific reliability improves performance in this setting.

## 6 Conclusion

We introduced RLSI, which formulates prompt-solvability estimation in group-relative RLVR as latent-variable inference from verifier rewards and response-specific reliability. Our analysis characterizes the limitations of direct reward aggregation, the information retained by response-level posterior inference, and the GRPO limit under perfect reliability. Across six reasoning benchmarks and two model scales, RLSI achieves the strongest average performance among the evaluated methods, while additional comparisons and ablations support the benefits of latent solvability inference and reliability heterogeneity.

## AI USE STATEMENT

In this work, we used generative AI tools to assist with English-language editing and manuscript organization, checking mathematical claims, writing and improving the exposition of proofs, providing feedback on experimental methodology and result interpretation, and implementing code. We did not use generative AI tools to develop the theoretical framework, formulate mathematical claims, provide critical ingredients for the proofs, propose or refine research hypotheses, generate synthetic datasets, clean or reformat datasets, translate text, execute experiments, or conduct qualitative or thematic data analysis; the latter task is not applicable to this work. All AI-assisted mathematical arguments were independently verified by the authors, suggested references were checked against their original sources, AI-assisted code was reviewed and tested by the authors, and experimental interpretations were validated against the actual results. All research decisions, method designs, final technical claims, proofs, implementations, and conclusions were reviewed and approved by the authors. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

## REFERENCES

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.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Peixuan Han, Adit Krishnan, Gerald Friedland, Jiaxuan You, and Luyang Kong. Self-aligned reward: Towards effective and efficient reasoners. In *International Conference on Learning Representations*, volume 2026, pp. 155846–155871, 2026.

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.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *Advances in Neural Information Processing Systems*, 38:162239–162262, 2026.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, et al. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*, 2024.

MAA. American mathematics competitions (amc). `https://maa.org/`, 2023.

MAA. American invitational mathematics examination (aime). `https://maa.org/`, 2025.

Youssef Mroueh. Reinforcement learning with verifiable rewards: Grpo's effective loss, dynamics, and success amplification. *arXiv preprint arXiv:2503.06639*, 2025.

Mihir Prabhudesai, Lili Chen, Alex Ippoliti, Katerina Fragkiadaki, Hao Liu, and Deepak Pathak. Maximizing confidence alone improves reasoning. *arXiv preprint arXiv:2505.22660*, 2025.

Ali Rad, Khashayar Filom, Darioush Keivan, Peyman Mohajerin Esfahani, and Ehsan Kamalinejad. Rate or fate? rlv$^{\varepsilon}$r: Reinforcement learning with verifiable noisy rewards, 2026. URL `https://arxiv.org/abs/2601.04411`.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for llm reasoning. In *International Conference on Learning Representations*, volume 2025, pp. 60808–60838, 2025.

Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, et al. Spurious rewards: Rethinking training signals in rlvr. *arXiv preprint arXiv:2506.10947*, 2025.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, 2025.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, 2024.

Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Liyuan Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, et al. Reinforcement learning for reasoning in large language models with one training example. *Advances in Neural Information Processing Systems*, 38: 122721–122764, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *Advances in Neural Information Processing Systems*, 38:113222–113244, 2026.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv preprint arXiv:2503.18892*, 2025.

Bonan Zhang, Zhongqi Chen, Bowen Song, Qinya Li, Fan Wu, and Guihai Chen. Confclip: Confidence-weighted and clipped reward for reinforcement learning in llms. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1981–1985. IEEE, 2026.

Xuandong Zhao, Zhewei Kang, Aosong Feng, Sergey Levine, and Dawn Song. Learning to reason without external rewards. In *International Conference on Learning Representations*, volume 2026, pp. 2548–2581, 2026.

Zhanke Zhou, Xiangyu Lu, Chentao Cao, Brando Miranda, Tongliang Liu, Bo Han, and Sanmi Koyejo. The easy, the hard, and the learnable: Confidence and difficulty-adaptive policy optimization for llm reasoning. *arXiv preprint arXiv:2606.07950*, 2026.

# A Related Works

**Reinforcement Learning with Verifiable Rewards.** Reinforcement learning with verifiable rewards (RLVR) improves reasoning policies using automatically verified outcome feedback, building on prior work on outcome-based supervision, process-based supervision, and learned verifiers for mathematical reasoning (Cobbe et al., 2021; Lightman et al., 2024; Wang et al., 2024; Uesato et al., 2022; Luo et al., 2024; Setlur et al., 2025). DeepSeekMath introduced Group Relative Policy Optimization (GRPO), which estimates response advantages from rewards within a sampled group without training a separate value model (Shao et al., 2024). DeepSeek-R1 subsequently demonstrated that large-scale RLVR can elicit reasoning behaviors such as verification, reflection, and strategy adaptation (Guo et al., 2025). The effectiveness and reproducibility of reasoning-oriented RL have subsequently been investigated across different model families, training scales, and data regimes (Team et al., 2025; Hu et al., 2026; Zeng et al., 2025; Wang et al., 2026). Later studies have improved the optimization stability and efficiency of this paradigm. DAPO introduces decoupled clipping and dynamic sampling to stabilize large-scale reasoning training (Yu et al., 2026), whereas Dr. GRPO analyzes optimization biases in the original GRPO formulation and proposes a corrected objective (Liu et al., 2025). A complementary line of work addresses the coarse credit assignment induced by trajectory-level rewards. Process reward models provide step-level supervision (Lightman et al., 2024; Wang et al., 2024). Recent analyses have further shown that RLVR behavior can be sensitive to the semantics and reliability of the verification signal, motivating closer examination of how verifier observations should enter policy optimization (Shao et al., 2025; Rad et al., 2026). These studies improve the optimization objective, sampling strategy, or temporal allocation of reward. RLSI instead addresses a different source of error in group-relative learning: the shared prompt-level baseline is commonly computed from the unweighted number of successful responses, even when individual verifier outcomes carry heterogeneous reliability. Our method therefore focuses on estimating prompt solvability from the joint reward–reliability evidence of the sampled group.

**Model-Derived Signals for Reasoning Optimization.** Recent methods use signals derived from the policy itself to supplement or replace sparse verifier feedback. RENT minimizes predictive entropy and treats model confidence as an intrinsic learning signal without requiring external correctness rewards (Prabhudesai et al., 2025), while Intuitor uses self-certainty as the reward for reasoning-policy optimization (Zhao et al., 2026). Other methods combine model-derived signals with verifier feedback. ConfClip (Zhang et al., 2026) transforms response rewards according to model confidence and clips the resulting signal to control unstable updates. Self-Aligned Reward measures how strongly a reasoning trajectory supports its final answer through a relative-perplexity signal and uses this quantity to complement outcome rewards (Han et al., 2026). CoDaPO combines rollout confidence with empirical prompt difficulty to adjust prompt selection and optimization emphasis during training (Zhou et al., 2026). Although these methods use model-derived scores to enrich the learning signal, the scores primarily operate as intrinsic rewards, reward transformations, prompt-selection criteria, or token-level credit weights. RLSI instead introduces response-specific reliability as an input to a latent-variable observation model, where it controls how an observed verifier outcome informs the latent response state. EM then aggregates the resulting response-level evidence into a shared estimate of prompt solvability for relative-advantage construction.

# B Proofs

*Proof of Proposition 4.1.* Because $R_{ij}$ is binary, its conditional expectation equals the probability of a positive verifier outcome. Marginalizing over $Z_{ij}$ under the latent solvability and observation

models gives

$$\begin{aligned}\mathbb{E}[R_{ij} \mid q_i, c_{ij}] &= P(R_{ij}=1 \mid q_i, c_{ij}) \\ &= P(R_{ij}=1 \mid Z_{ij}=1, c_{ij})P(Z_{ij}=1 \mid q_i) \\ &\quad + P(R_{ij}=1 \mid Z_{ij}=0, c_{ij})P(Z_{ij}=0 \mid q_i) \\ &= c_{ij}q_i + (1-c_{ij})(1-q_i) \\ &= (1-c_{ij}) + (2c_{ij}-1)q_i.\end{aligned}$$

For the group-average estimator, linearity of conditional expectation and $\bar{c}_i = G^{-1}\sum_{j=1}^{G} c_{ij}$ yield

$$\begin{aligned}\mathbb{E}[\widehat{q}_i^{\mathrm{GA}} \mid q_i, \mathbf{c}_i] - q_i &= \frac{1}{G}\sum_{j=1}^{G}\mathbb{E}[R_{ij} \mid q_i, c_{ij}] - q_i \\ &= \frac{1}{G}\sum_{j=1}^{G}[(1-c_{ij}) + (2c_{ij}-1)q_i] - q_i \\ &= (1-\bar{c}_i) + (2\bar{c}_i - 1)q_i - q_i \\ &= (1-\bar{c}_i)(1-2q_i).\end{aligned}$$

For the reliability-weighted estimator, conditioning on $\mathbf{c}_i$ makes its denominator fixed. Therefore,

$$\begin{aligned}\mathbb{E}[\widehat{q}_i^{\mathrm{RW}} \mid q_i, \mathbf{c}_i] - q_i &= \frac{\sum_{j=1}^{G} c_{ij}\mathbb{E}[R_{ij} \mid q_i, c_{ij}]}{\sum_{j=1}^{G} c_{ij}} - q_i \\ &= \frac{\sum_{j=1}^{G} c_{ij}\left[(1-c_{ij}) + (2c_{ij}-1)q_i\right] - q_i\sum_{j=1}^{G} c_{ij}}{\sum_{j=1}^{G} c_{ij}} \\ &= \frac{\sum_{j=1}^{G} c_{ij}(1-c_{ij}) + q_i\sum_{j=1}^{G}(2c_{ij}^2 - 2c_{ij})}{\sum_{j=1}^{G} c_{ij}} \\ &= \frac{\sum_{j=1}^{G} c_{ij}(1-c_{ij}) - 2q_i\sum_{j=1}^{G} c_{ij}(1-c_{ij})}{\sum_{j=1}^{G} c_{ij}} \\ &= (1-2q_i)\frac{\sum_{j=1}^{G} c_{ij}(1-c_{ij})}{\sum_{j=1}^{G} c_{ij}}.\end{aligned}$$

Since $c_{ij} \in (0,1)$ for every $j$, both $1-\bar{c}_i$ and $\sum_{j=1}^{G} c_{ij}(1-c_{ij})/\sum_{j=1}^{G} c_{ij}$ are strictly positive. Hence, both conditional biases are nonzero whenever $q_i \neq 1/2$. □

*Proof of Theorem 4.2.* Fix $\mathbf{c}_i$ and $q_i^{(k)}$. For each response, define

$$\begin{aligned}a_{ij} &= P(R_{ij} \mid Z_{ij}=1, c_{ij}) = c_{ij}^{R_{ij}}(1-c_{ij})^{1-R_{ij}}, \\ b_{ij} &= P(R_{ij} \mid Z_{ij}=0, c_{ij}) = (1-c_{ij})^{R_{ij}}c_{ij}^{1-R_{ij}}.\end{aligned}$$

Here $a_{ij}$ and $b_{ij}$ are the probabilities of the observed reward under the two possible latent states.

For any $\mathbf{z} = (z_1, \ldots, z_G) \in \{0,1\}^G$, the conditional independence structure of the model gives

$$\begin{aligned}&P(\mathbf{Z}_i = \mathbf{z}, \mathbf{R}_i \mid \mathbf{c}_i, q_i^{(k)}) \\ &\quad= \prod_{j=1}^{G} P(Z_{ij}=z_j \mid q_i^{(k)}, c_{ij})P(R_{ij} \mid Z_{ij}=z_j, c_{ij}) \\ &\quad= \prod_{j=1}^{G}\left(q_i^{(k)}a_{ij}\right)^{z_j}\left((1-q_i^{(k)})b_{ij}\right)^{1-z_j}.\end{aligned}$$

We obtain the probability of the observed reward vector by summing this joint probability over all possible latent-state vectors. Because the joint probability is a product across responses, the sum factorizes:

$$P(\mathbf{R}_i \mid \mathbf{c}_i, q_i^{(k)}) = \sum_{\mathbf{z}\in\{0,1\}^G} P(\mathbf{Z}_i = \mathbf{z}, \mathbf{R}_i \mid \mathbf{c}_i, q_i^{(k)}) = \prod_{j=1}^{G} \left[ q_i^{(k)} a_{ij} + (1 - q_i^{(k)}) b_{ij} \right].$$

Dividing the joint probability by this marginal probability yields the posterior:

$$P(\mathbf{Z}_i = \mathbf{z} \mid \mathbf{R}_i, \mathbf{c}_i, q_i^{(k)}) = \prod_{j=1}^{G} \left( \frac{q_i^{(k)} a_{ij}}{q_i^{(k)} a_{ij} + (1 - q_i^{(k)}) b_{ij}} \right)^{z_j} \times \left( \frac{(1 - q_i^{(k)}) b_{ij}}{q_i^{(k)} a_{ij} + (1 - q_i^{(k)}) b_{ij}} \right)^{1 - z_j}.$$

By the E-step definition in equation 3.6, the first fraction in each factor is $\gamma_{ij}^{(k)}$. The second fraction is $1 - \gamma_{ij}^{(k)}$. Therefore,

$$P(\mathbf{Z}_i = \mathbf{z} \mid \mathbf{R}_i, \mathbf{c}_i, q_i^{(k)}) = \prod_{j=1}^{G} \left(\gamma_{ij}^{(k)}\right)^{z_j} \left(1 - \gamma_{ij}^{(k)}\right)^{1 - z_j},$$

which proves equation 4.2.

We now establish the conditional-independence statement. For fixed $\mathbf{c}_i$ and $q_i^{(k)}$, $\boldsymbol{\gamma}_i^{(k)}$ is a deterministic function of $\mathbf{R}_i$. The factorization above shows that $P(\mathbf{Z}_i = \mathbf{z} \mid \mathbf{R}_i, \mathbf{c}_i, q_i^{(k)})$ depends on $\mathbf{R}_i$ only through this function. Applying the law of iterated expectation gives

$$P(\mathbf{Z}_i = \mathbf{z} \mid \boldsymbol{\gamma}_i^{(k)}, \mathbf{c}_i, q_i^{(k)}) = \mathbb{E}\left[ P(\mathbf{Z}_i = \mathbf{z} \mid \mathbf{R}_i, \mathbf{c}_i, q_i^{(k)}) \;\middle|\; \boldsymbol{\gamma}_i^{(k)}, \mathbf{c}_i, q_i^{(k)} \right] = \prod_{j=1}^{G} \left(\gamma_{ij}^{(k)}\right)^{z_j} \left(1 - \gamma_{ij}^{(k)}\right)^{1 - z_j}.$$

Conditioning additionally on $\mathbf{R}_i$ gives the same expression, as established above. Hence $\mathbf{Z}_i \perp \mathbf{R}_i \mid \boldsymbol{\gamma}_i^{(k)}, \mathbf{c}_i, q_i^{(k)}$. □

*Proof of Theorem 4.4.* Fix a response $j$ and write $q = q_i^{(k)}$, $c = c_{ij}$, and $R = R_{ij}$ within this proof. Define the likelihoods of the observed reward under the two latent states by

$$a = P(R \mid Z_{ij} = 1, c) = c^R (1 - c)^{1 - R}, \qquad b = P(R \mid Z_{ij} = 0, c) = (1 - c)^R c^{1 - R}.$$

The E-step posterior and its complement are

$$\gamma_{ij}^{(k)} = \frac{qa}{qa + (1 - q)b}, \qquad 1 - \gamma_{ij}^{(k)} = \frac{(1 - q)b}{qa + (1 - q)b}.$$

Dividing the first expression by the second cancels their common denominator:

$$\frac{\gamma_{ij}^{(k)}}{1 - \gamma_{ij}^{(k)}} = \frac{q}{1 - q} \frac{a}{b}.$$

The likelihood ratio can be simplified without omitting the dependence on the observed reward:

$$\begin{aligned}\frac{a}{b} &= \frac{c^R(1-c)^{1-R}}{(1-c)^R c^{1-R}} \\ &= c^{R-(1-R)}(1-c)^{(1-R)-R} \\ &= \left(\frac{c}{1-c}\right)^{2R-1}.\end{aligned}$$

Substituting this ratio into the posterior odds and taking logarithms gives

$$\begin{aligned}\operatorname{logit}\gamma_{ij}^{(k)} &= \log\frac{\gamma_{ij}^{(k)}}{1-\gamma_{ij}^{(k)}} \\ &= \log\frac{q}{1-q} + (2R-1)\log\frac{c}{1-c} \\ &= \operatorname{logit} q_i^{(k)} + (2R_{ij}-1)\operatorname{logit} c_{ij}.\end{aligned}$$

This proves the signed reliability identity stated in Theorem 4.4.

To establish the stated directions, first note that $\operatorname{logit} c_{ij}$ is zero at $c_{ij} = 1/2$, positive when $c_{ij} > 1/2$, and negative when $c_{ij} < 1/2$. If $R_{ij} = 1$, then $2R_{ij} - 1 = 1$, so the posterior log-odds differ from the prior log-odds by $\operatorname{logit} c_{ij}$. If $R_{ij} = 0$, then $2R_{ij} - 1 = -1$, so they differ by $-\operatorname{logit} c_{ij}$. Because the logit function is strictly increasing on $(0, 1)$, these comparisons give exactly the relationships between $\gamma_{ij}^{(k)}$ and $q_i^{(k)}$ stated in the theorem.

Finally, keep $q = q_i^{(k)} \in (0, 1)$ fixed and let $c = c_{ij} \to 1$. For a positive reward, the E-step becomes

$$\gamma_{ij}^{(k)} = \frac{qc}{qc + (1-q)(1-c)} \longrightarrow 1 \qquad \text{if } R_{ij} = 1.$$

For a negative reward, it becomes

$$\gamma_{ij}^{(k)} = \frac{q(1-c)}{q(1-c) + (1-q)c} \longrightarrow 0 \qquad \text{if } R_{ij} = 0.$$

Thus $\gamma_{ij}^{(k)} \to R_{ij}$ in both cases. When $c_{ij} \to 1$ for every response, the M-step gives

$$q_i^{(k+1)} = \frac{1}{G}\sum_{j=1}^{G}\gamma_{ij}^{(k)} \longrightarrow \frac{1}{G}\sum_{j=1}^{G} R_{ij} = \bar{R}_i.$$

This proves the perfect-reliability limit stated in Theorem 4.4. □

*Proof of Theorem 4.5.* Write $q = q_i^{(k)}$ for brevity. For each response, define

$$\begin{aligned}a_{ij} &= P(R_{ij} \mid Z_{ij} = 1, c_{ij}) = c_{ij}^{R_{ij}}(1-c_{ij})^{1-R_{ij}}, \\ b_{ij} &= P(R_{ij} \mid Z_{ij} = 0, c_{ij}) = (1-c_{ij})^{R_{ij}} c_{ij}^{1-R_{ij}}.\end{aligned}$$

The marginal likelihood of its observed reward is $m_{ij}^{(k)} = qa_{ij} + (1-q)b_{ij}$, and the E-step posterior is $\gamma_{ij}^{(k)} = qa_{ij}/m_{ij}^{(k)}$. Hence the difference between the response-level posterior and the current prompt-level parameter is

$$\begin{aligned}\gamma_{ij}^{(k)} - q &= \frac{qa_{ij}}{m_{ij}^{(k)}} - q \\ &= \frac{qa_{ij} - q\{qa_{ij} + (1-q)b_{ij}\}}{m_{ij}^{(k)}} \\ &= \frac{qa_{ij} - q^2 a_{ij} - q(1-q)b_{ij}}{m_{ij}^{(k)}} \\ &= \frac{q(1-q)(a_{ij} - b_{ij})}{m_{ij}^{(k)}}.\end{aligned}$$

We now express $a_{ij} - b_{ij}$ in terms of the observed reward and reliability. If $R_{ij} = 1$, then $a_{ij} = c_{ij}$ and $b_{ij} = 1 - c_{ij}$, so $a_{ij} - b_{ij} = 2c_{ij} - 1$. If $R_{ij} = 0$, then $a_{ij} = 1 - c_{ij}$ and $b_{ij} = c_{ij}$, so $a_{ij} - b_{ij} = -(2c_{ij} - 1)$. The two cases combine as

$$a_{ij} - b_{ij} = (2R_{ij} - 1)(2c_{ij} - 1).$$

Substitution gives the correction contributed by response $j$:

$$\gamma_{ij}^{(k)} - q_i^{(k)} = \frac{q_i^{(k)}(1 - q_i^{(k)})(2R_{ij} - 1)(2c_{ij} - 1)}{m_{ij}^{(k)}}.$$

The M-step sets $q_i^{(k+1)} = G^{-1} \sum_{j=1}^{G} \gamma_{ij}^{(k)}$. Since $q_i^{(k)}$ is shared by all $G$ responses, we can write the change in the prompt-level estimate as the average of their individual corrections:

$$\begin{aligned} q_i^{(k+1)} - q_i^{(k)} &= \frac{1}{G} \sum_{j=1}^{G} \left( \gamma_{ij}^{(k)} - q_i^{(k)} \right) \\ &= \frac{q_i^{(k)}(1 - q_i^{(k)})}{G} \sum_{j=1}^{G} \frac{(2R_{ij} - 1)(2c_{ij} - 1)}{m_{ij}^{(k)}}. \end{aligned}$$

This is equation 4.4. □

*Proof of Corollary 4.3.* Let $C_i = (\mathbf{c}_i, q_i^{(k)})$ and write $\Gamma_i = \boldsymbol{\gamma}_i^{(k)}$ to shorten the conditioning notation. Recall that $S_i = \sum_{j=1}^{G} Z_{ij}$ is a deterministic function of $\mathbf{Z}_i$.

Theorem 4.2 establishes $\mathbf{Z}_i \perp \mathbf{R}_i \mid \Gamma_i, C_i$. Therefore, $S_i \perp \mathbf{R}_i \mid \Gamma_i, C_i$, and

$$I(S_i; \mathbf{R}_i \mid \Gamma_i, C_i) = 0.$$

For fixed $C_i$, the E-step makes $\Gamma_i$ a deterministic function of $\mathbf{R}_i$. Adding $\Gamma_i$ to $\mathbf{R}_i$ therefore adds no information once $\mathbf{R}_i$ and $C_i$ are given:

$$I(S_i; \Gamma_i \mid \mathbf{R}_i, C_i) = 0.$$

Apply the chain rule to the same joint quantity in two orders:

$$\begin{aligned} I(S_i; \mathbf{R}_i, \Gamma_i \mid C_i) &= I(S_i; \mathbf{R}_i \mid C_i) + I(S_i; \Gamma_i \mid \mathbf{R}_i, C_i) \\ &= I(S_i; \mathbf{R}_i \mid C_i), \end{aligned}$$

whereas

$$\begin{aligned} I(S_i; \Gamma_i, \mathbf{R}_i \mid C_i) &= I(S_i; \Gamma_i \mid C_i) + I(S_i; \mathbf{R}_i \mid \Gamma_i, C_i) \\ &= I(S_i; \Gamma_i \mid C_i). \end{aligned}$$

The left-hand sides are the same, proving

$$I(S_i; \Gamma_i \mid C_i) = I(S_i; \mathbf{R}_i \mid C_i).$$

Next, $\bar{R}_i = G^{-1} \sum_{j=1}^{G} R_{ij}$ is a deterministic function of $\mathbf{R}_i$. By the same chain-rule argument,

$$\begin{aligned} I(S_i; \mathbf{R}_i \mid C_i) &= I(S_i; \bar{R}_i, \mathbf{R}_i \mid C_i) \\ &= I(S_i; \bar{R}_i \mid C_i) + I(S_i; \mathbf{R}_i \mid \bar{R}_i, C_i). \end{aligned}$$

Conditional mutual information is nonnegative, so the last expression is at least $I(S_i; \bar{R}_i \mid C_i)$. Together with the equality above, this proves equation 4.3. It also shows that the inequality is strict exactly when $I(S_i; \mathbf{R}_i \mid \bar{R}_i, C_i) > 0$.

□

## C Implementation Details

**Training configurations.** We implement RLSI and all baselines using the verl framework (Sheng et al., 2025). All methods are trained on DAPO-Math-17K (Yu et al., 2026) and initialized from the same checkpoint within each model scale, namely Qwen3-1.7B-Base or Qwen3-4B (Yang et al., 2025). For all methods, each rollout batch contains 128 prompts with $G = 4$ responses per prompt, yielding 512 trajectories. The maximum prompt and response lengths are 1024 and 2048 tokens, respectively. We use an actor learning rate of $10^{-6}$ and an actor minibatch size of 128. Unless otherwise specified below, the remaining optimizer settings follow the corresponding verl configurations. All models are trained for 400 optimization steps and evaluated using their final checkpoints. Each training run is conducted on a Slurm-managed cluster using eight NVIDIA A6000 GPUs. Our anonymous code is available at `https://anonymous.4open.science/r/RLSI-7505/`.

**Algorithm 1** Reliability-Aware Latent Solvability Inference

**Require:** Reference policy $\pi_{\text{ref}}$, initial policy $\pi_{\theta_0}$, verifier $\mathcal{V}$, prompt set $\mathcal{X} = \{x_i\}_{i=1}^{N}$, group size $G$, initial solvability value $q_0$, offline EM steps $K_{\text{off}}$, RL steps $T$, stability constant $\epsilon$

**Offline initialization**
**for** $i = 1, \dots, N$ **do**
  Sample $\{y_{ij}\}_{j=1}^{G} \sim \pi_{\text{ref}}(\cdot \mid x_i)$ and compute $R_{ij} = \mathcal{V}(x_i, y_{ij})$, reliability scores $c_{ij}$.
  $q_i \leftarrow q_0$.
  **for** $k = 1, \dots, K_{\text{off}}$ **do**
    $\gamma_{ij} \leftarrow \Gamma(R_{ij}, c_{ij}; q_i), \quad q_i \leftarrow G^{-1} \sum_{j=1}^{G} \gamma_{ij}$.
  **end for**
  $q_i^{(0)} \leftarrow q_i$.
**end for**
**Online policy optimization**
**for** $t = 1, \dots, T$ **do**
  Sample a prompt minibatch $\mathcal{B}_t$.
  **for** each $i \in \mathcal{B}_t$ **do**
    Sample $\{y_{ij}^{(t)}\}_{j=1}^{G} \sim \pi_{\theta_{t-1}}(\cdot \mid x_i)$ and compute $R_{ij}^{(t)}$ and $c_{ij}^{(t)}$.
    **One-step EM update:**

$$\gamma_{ij}^{(t)} \leftarrow \Gamma\left(R_{ij}^{(t)}, c_{ij}^{(t)}; q_i^{(t-1)}\right), \qquad q_i^{(t)} \leftarrow \frac{1}{G} \sum_{j=1}^{G} \gamma_{ij}^{(t)}.$$

    Compute

$$A_{ij}^{(t)} = \frac{R_{ij}^{(t)} - q_i^{(t)}}{\sqrt{q_i^{(t)}(1 - q_i^{(t)}) + \epsilon}}.$$

  **end for**
  Update $\pi_{\theta_{t-1}} \to \pi_{\theta_t}$ using GRPO with $\{A_{ij}^{(t)}\}$.
**end for**

**RLSI and Baseline hyperparameters.** For PPO, GRPO, and DAPO, we use the corresponding default recipes provided by verl (Sheng et al., 2025). RW-GRPO follows the GRPO configuration and differs only in its prompt-level baseline construction. RLSI follows the default GRPO recipe except for the method-specific settings introduced by solvability estimation and reliability-adjusted advantage construction. For offline prompt-specific initialization, we initialize the solvability estimate at $q_0 = 0.5$, and run EM for 30 iterations.

**Evaluation configurations.** We evaluate all methods on the same six mathematical reasoning benchmarks using an identical answer-verification procedure. For MATH500, Minerva Math, and OlympiadBench, we generate one response per problem and report Avg@1. For AIME24, AIME25, and AMC23, we generate 32 responses per problem and report Avg@32. The overall score is the unweighted arithmetic mean of the six benchmark accuracies. We use temperature 0 for Avg@1 evaluation and temperature 1 for Avg@32 evaluation, with random seed 42 in both settings.

# D QUALITATIVE CASE STUDY

We present three representative MATH500 problems for which GRPO produces an incorrect answer whereas RLSI produces the correct answer. The examples cover algebraic reasoning, number-theoretic computation, and combinatorial counting. The responses are reproduced step by step below.

## D.1 CASE 1: A NESTED-RADICAL EQUATION

**Problem**

Solve

$$\sqrt{x+\sqrt{3x+6}}+\sqrt{x-\sqrt{3x+6}}=6.$$

**Reference answer:** 10.

**GRPO Response**

To solve the equation $\sqrt{x+\sqrt{3x+6}}+\sqrt{x-\sqrt{3x+6}}=6$, we can follow these steps:

1. **Isolate one of the square roots**
   Let us isolate $\sqrt{x+\sqrt{3x+6}}$:
   $$\sqrt{x+\sqrt{3x+6}}=6-\sqrt{x-\sqrt{3x+6}}.$$
2. **Square both sides**
   Square both sides of the equation to eliminate the square root on the left side:
   $$x+\sqrt{3x+6}=\left(6-\sqrt{x-\sqrt{3x+6}}\right)^2.$$
   Expanding the right side, we get:
   $$x+\sqrt{3x+6}=36-12\sqrt{x-\sqrt{3x+6}}+\left(x-\sqrt{3x+6}\right).$$
   Simplifying, we get:
   $$x+\sqrt{3x+6}=36-12\sqrt{x-\sqrt{3x+6}}+x-\sqrt{3x+6}.$$
   Cancel out $x$ from both sides:
   $$\sqrt{3x+6}=36-12\sqrt{x-\sqrt{3x+6}}-\sqrt{3x+6}.$$
   Combine like terms:
   $$2\sqrt{3x+6}=36-12\sqrt{x-\sqrt{3x+6}}.$$
   Divide both sides by 2:
   $$\sqrt{3x+6}=18-6\sqrt{x-\sqrt{3x+6}}.$$
3. **Isolate the remaining square root**
   Let us isolate $\sqrt{x-\sqrt{3x+6}}$:
   $$\sqrt{x-\sqrt{3x+6}}=18-\sqrt{3x+6}.$$

4. **Square both sides again**
   Square both sides to eliminate the square root on the left side:
   $$x - \sqrt{3x+6} = \left(18 - \sqrt{3x+6}\right)^2.$$
   Expanding the right side, we get:
   $$x - \sqrt{3x+6} = 324 - 36\sqrt{3x+6} + (3x+6).$$
   Simplifying, we get:
   $$x - \sqrt{3x+6} = 330 + 3x - 36\sqrt{3x+6}.$$
   Cancel out $x$ from both sides:
   $$-\sqrt{3x+6} = 330 + 2x - 36\sqrt{3x+6}.$$
   Combine like terms:
   $$35\sqrt{3x+6} = 330 + 2x.$$
   Divide both sides by 35:
   $$\sqrt{3x+6} = \frac{330+2x}{35}.$$
5. **Square both sides again**
   Square both sides to eliminate the square root:
   $$3x + 6 = \left(\frac{330+2x}{35}\right)^2.$$
   Simplifying, we get:
   $$3x + 6 = \frac{(330+2x)^2}{1225}.$$
   Multiply both sides by 1225:
   $$1225(3x+6) = (330+2x)^2.$$
   Expanding the right side, we get:
   $$3675x + 7350 = 108900 + 1320x + 4x^2.$$
   Rearrange all terms to one side:
   $$4x^2 + 1320x + 108900 - 3675x - 7350 = 0.$$
   Simplify:
   $$4x^2 - 2355x + 101550 = 0.$$
   Divide the entire equation by 4:
   $$x^2 - 588.75x + 25387.5 = 0.$$
6. **Solve the quadratic equation**
   Use the quadratic formula $x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$, where $a = 1$, $b = -588.75$, and $c = 25387.5$:
   $$x = \frac{588.75 \pm \sqrt{588.75^2 - 4 \cdot 1 \cdot 25387.5}}{2 \cdot 1}.$$
   Simplifying, we get:
   $$x = \frac{588.75 \pm \sqrt{346326.5625 - 101550}}{2},$$

$$x = \frac{588.75 \pm \sqrt{244776.5625}}{2},$$

and

$$x = \frac{588.75 \pm 494.75}{2}.$$

This gives us two solutions:

$$x = \frac{588.75 + 494.75}{2} = 541.75$$

and

$$x = \frac{588.75 - 494.75}{2} = 47.$$

7. **Check the solutions**
   Substitute $x = 47$ and $x = 541.75$ back into the original equation to check which one satisfies it. We find that $x = 47$ is the valid solution.

Therefore, the final answer is

$$\boxed{47}.$$

**RLSI Response**

To solve the equation $\sqrt{x + \sqrt{3x + 6}} + \sqrt{x - \sqrt{3x + 6}} = 6$, we will proceed step by step.

**Step 1: Simplify the equation**
Let us denote

$$a = \sqrt{x + \sqrt{3x + 6}}$$

and

$$b = \sqrt{x - \sqrt{3x + 6}}.$$

The given equation is

$$a + b = 6.$$

**Step 2: Square both sides**
To eliminate the square roots, we square both sides of $a + b = 6$:

$$(a + b)^2 = 6^2,$$

so

$$a^2 + b^2 + 2ab = 36.$$

**Step 3: Express $a^2$ and $b^2$**
We know that

$$a^2 = x + \sqrt{3x + 6}$$

and

$$b^2 = x - \sqrt{3x + 6}.$$

Adding these two equations gives

$$a^2 + b^2 = \left(x + \sqrt{3x + 6}\right) + \left(x - \sqrt{3x + 6}\right),$$

and hence

$$a^2 + b^2 = 2x.$$

**Step 4: Substitute $a^2 + b^2$ into the squared equation**
From Step 2, we have

$$a^2 + b^2 + 2ab = 36.$$

Substituting $a^2 + b^2 = 2x$ gives

$$2x + 2ab = 36,$$

$$2ab = 36 - 2x,$$

and therefore

$$ab = 18 - x.$$

**Step 5: Express $ab$**
We know that

$$a = \sqrt{x + \sqrt{3x + 6}}$$

and

$$b = \sqrt{x - \sqrt{3x + 6}}.$$

Thus,

$$ab = \sqrt{\left(x + \sqrt{3x + 6}\right)\left(x - \sqrt{3x + 6}\right)}.$$

**Step 6: Simplify the product inside the square root**
Using the difference-of-squares identity,

$$\left(x + \sqrt{3x + 6}\right)\left(x - \sqrt{3x + 6}\right) = x^2 - \left(\sqrt{3x + 6}\right)^2,$$
$$= x^2 - (3x + 6),$$

and hence

$$= x^2 - 3x - 6.$$

Therefore,

$$ab = \sqrt{x^2 - 3x - 6}.$$

**Step 7: Equate the two expressions for $ab$**
From Step 4, we have $ab = 18 - x$. Therefore,

$$\sqrt{x^2 - 3x - 6} = 18 - x.$$

**Step 8: Square both sides again**
To eliminate the square root, we square both sides:

$$x^2 - 3x - 6 = (18 - x)^2,$$

which gives

$$x^2 - 3x - 6 = 324 - 36x + x^2.$$

**Step 9: Simplify the equation**
Subtracting $x^2$ from both sides gives

$$-3x - 6 = 324 - 36x.$$

Adding $36x$ to both sides gives

$$33x - 6 = 324.$$

Adding 6 to both sides gives

$$33x = 330.$$

Dividing by 33, we obtain

$$x = 10.$$

**Step 10: Verify the solution**
Substituting $x = 10$ back into the original equation gives

$$\sqrt{10 + \sqrt{3(10) + 6}} + \sqrt{10 - \sqrt{3(10) + 6}} = 6,$$
$$\sqrt{10 + \sqrt{30 + 6}} + \sqrt{10 - \sqrt{30 + 6}} = 6,$$
$$\sqrt{10 + \sqrt{36}} + \sqrt{10 - \sqrt{36}} = 6,$$
$$\sqrt{10 + 6} + \sqrt{10 - 6} = 6,$$
$$\sqrt{16} + \sqrt{4} = 6,$$

and

$$4 + 2 = 6.$$

The solution is therefore verified. The final answer is

$$\boxed{10}.$$

### D.2 Case 2: Computing a Greatest Common Factor

**Problem**

What is the result when the greatest common factor of 6432 and 132 is increased by 11?

**Reference answer:** 23.

**GRPO Response**

To find the greatest common factor (GCF) of 6432 and 132, we can use the Euclidean algorithm. The Euclidean algorithm involves repeated division and taking remainders until we reach a remainder of 0. The last nonzero remainder is the GCF.

1. Divide 6432 by 132 and find the remainder:

$$6432 \div 132 = 48 \qquad \text{remainder } 0.$$

   Since the remainder is 0, the GCF of 6432 and 132 is 132.

2. Now, we need to increase this GCF by 11:

$$132 + 11 = 143.$$

Therefore, the final answer is

$$\boxed{143}.$$

**RLSI Response**

To solve the problem, we need to follow these steps:

1. Find the greatest common factor (GCF) of 6432 and 132.
2. Increase the GCF by 11.
3. Output the final answer.

**Step 1: Find the GCF of** 6432 **and** 132
To find the GCF, we can use the Euclidean algorithm. The Euclidean algorithm involves repeated division and taking remainders until we reach a remainder of 0. The last nonzero remainder is the GCF.

**Euclidean Algorithm Steps**

1. Divide 6432 by 132 and find the remainder:

   $6432 \div 132 = 48$ (quotient), $6432 - 48 \times 132 = 6432 - 6336 = 96$ (remainder).

   Thus,

$$6432 = 132 \times 48 + 96.$$

2. Now divide 132 by 96 and find the remainder:

   $132 \div 96 = 1$ (quotient), $132 - 1 \times 96 = 132 - 96 = 36$ (remainder).

   Thus,

$$132 = 96 \times 1 + 36.$$

3. Next, divide 96 by 36 and find the remainder:

   $96 \div 36 = 2$ (quotient), $96 - 2 \times 36 = 96 - 72 = 24$ (remainder).

   Thus,

$$96 = 36 \times 2 + 24.$$

4. Now divide 36 by 24 and find the remainder:

   $36 \div 24 = 1$ (quotient), $36 - 1 \times 24 = 36 - 24 = 12$ (remainder).

   Thus,

$$36 = 24 \times 1 + 12.$$

5. Finally, divide 24 by 12 and find the remainder:

$$24 \div 12 = 2 \quad \text{(quotient)}, \qquad 24 - 2 \times 12 = 24 - 24 = 0 \quad \text{(remainder)}.$$

Thus,

$$24 = 12 \times 2 + 0.$$

Since the remainder is now 0, the last nonzero remainder is the GCF. Therefore, the GCF of 6432 and 132 is

$$\boxed{12}.$$

**Step 2: Increase the GCF by** 11
Now, we need to increase the GCF by 11:

$$12 + 11 = 23.$$

**Step 3: Output the Final Answer**
The final answer is

$$\boxed{23}.$$

### D.3 Case 3: Indistinguishable Balls and Boxes

**Problem**

How many ways are there to put 5 balls in 2 boxes if the balls are not distinguishable and neither are the boxes?

**Reference answer:** 3.

**GRPO Response**

To solve this problem, we need to determine the number of ways to distribute 5 indistinguishable balls into 2 indistinguishable boxes. This is a classic combinatorial problem that can be solved using the concept of partitions of a number.

**Step-by-Step Reasoning**

1. **Understanding the Problem**
   - We have 5 indistinguishable balls.
   - We have 2 indistinguishable boxes.
   - We need to find the number of ways to distribute the balls into the boxes.
2. **Possible Distributions**
   Since the balls and boxes are indistinguishable, the order of the boxes does not matter. We only need to consider the number of balls in each box.
3. **Possible Cases**
   - **Case 1:** All 5 balls in one box.
     This is the only way to distribute the balls if one box is empty.
   - **Case 2:** 4 balls in one box and 1 ball in the other box.
     This is another valid distribution.
   - **Case 3:** 3 balls in one box and 2 balls in the other box.
     This is another valid distribution.
   - **Case 4:** 2 balls in one box and 3 balls in the other box.
     This is another valid distribution.
   - **Case 5:** 1 ball in one box and 4 balls in the other box.
     This is another valid distribution.
   - **Case 6:** 0 balls in one box and 5 balls in the other box.
     This is another valid distribution.

4. **Counting the Cases**
   There are 6 distinct ways to distribute the balls into the boxes.

Therefore, the final answer is

$$\boxed{6}.$$

**RLSI Response**

To solve the problem of determining how many ways there are to put 5 indistinguishable balls into 2 indistinguishable boxes, we need to consider the different possible distributions of the balls between the boxes. Since the balls and the boxes are indistinguishable, the order of the boxes does not matter, and we only need to count unique distributions.

**Step 1: Identify the possible distributions**
We need to find all possible ways to distribute 5 indistinguishable balls into 2 indistinguishable boxes. The possible distributions are:

1. All 5 balls in one box and 0 balls in the other box.
2. 4 balls in one box and 1 ball in the other box.
3. 3 balls in one box and 2 balls in the other box.

These are the only possible distributions because any other distribution would either repeat a previous one or exceed the total number of balls.

**Step 2: List the unique distributions**
The unique distributions are

$$(5,0), \qquad (4,1), \qquad (3,2).$$

**Step 3: Count the unique distributions**
We have listed all the unique distributions, and there are no other possible distributions. Therefore, the number of ways to distribute the balls is the number of unique distributions identified above.

**Final Answer**
The final answer is

$$\boxed{3}.$$