DATALENS: PREDICTING FINITE-STEP DATA UTILITY FOR DATA-EFFICIENT LLM REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Selecting effective training data is crucial for efficient reinforcement learning (RL) of large language models (LLMs), yet example utility cannot be reliably inferred from local training signals. Existing gradient-based methods use local gradient information as a proxy for training improvement. We show that this approximation can break down for finite displacements: candidates with similar local alignment may produce substantially different target-objective improvements. We formulate finite-step data utility prediction and define utility as the target-objective improvement induced by a prescribed candidate-specific update. We propose DataLens, which augments first-order alignment with directional objective geometry derived from a finite-step expansion, modeling objective evolution along candidate-induced update directions. Controlled matched-objective experiments verify that this geometric signal captures utility differences missed by local approximations. Across six model-domain settings spanning Qwen, DeepSeek, and Mistral on math and code, DataLens achieves the highest point-estimate correlation with finite-step utility and transfers to short-horizon GRPO. In end-to-end GRPO experiments, 100 examples selected by DataLens achieve 84.0% MATH accuracy across three random seeds, compared with 70.1% using first-order alignment and 67.5% using random selection. These results show that RL data selection benefits from modeling finite-step objective geometry beyond local gradient signals.

1 INTRODUCTION

Reinforcement learning (RL) has become an important component of large language model (LLM) post-training, driving improvements in mathematical reasoning, code generation, and other tasks with verifiable rewards (Shao et al., 2024; DeepSeek-AI, 2025). As these training pipelines scale, however, the available optimization budget remains limited. This makes the choice of training examples increasingly important: different examples can require similar rollout and optimization cost while producing very different improvements in the resulting model. A natural approach is to estimate example value from local training signals. Gradient alignment, gradient norm, difficulty, and influence-based scores all provide low-cost signals at the current model state, and have been widely used as proxies for downstream training utility. These methods implicitly rely on the assumption that a candidate that looks favorable locally will also produce a favorable change after optimization. For infinitesimal updates, this approximation is well justified. In practical reinforcement learning, however, updates are finite, and the target objective may change nonlinearly along the direction induced by a candidate.

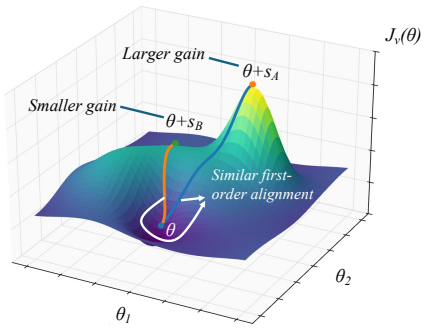

Figure 1: **Finite-step data utility.** Candidate displacements from the same parameter state yield different changes in the target objective.

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.

This distinction matters because local alignment alone does not determine the outcome of a finite update. Consider two candidates whose update directions have similar first-order alignment with a target objective. If the target objective has different curvature along these directions, the same local signal can lead to substantially different finite-step improvements. This suggests that effective data selection should model not only the gradient at the current point, but also how the target objective evolves along the update directions induced by candidate examples. Figure 1 illustrates this intuition. Starting from the same parameter state, two candidate-induced finite displacements can move the model to regions with substantially different target-objective values. Thus, the utility of a candidate is determined not only by the local signal at the current point, but also by how the target objective evolves along the update direction over a finite optimization step.

We therefore formulate the problem of *finite-step data utility prediction*: predicting the target-objective improvement induced by a candidate-specific update. Scoring candidates at a common model state and update scale provides a consistent basis for comparing their predicted gains under a shared training budget. Evaluating every candidate directly is expensive, so we ask whether directional objective geometry can improve this prediction beyond local gradient signals. To this end, we propose **DataLens**, a selection-time method that combines first-order target alignment with directional objective geometry to predict finite-step utility. DataLens does not modify the downstream RL algorithm. Instead, it scores candidates before training, ranks them by predicted utility, and trains standard GRPO on the selected subset. Our primary implementation computes this geometry in a projected LM-head subspace, where the required directional curvature can be evaluated efficiently without explicitly materializing the full Hessian.

We evaluate DataLens at three levels. First, we directly measure controlled finite-step utility and test whether DataLens predicts it more accurately than local and curvature-based alternatives. Second, we test whether the resulting scores transfer to realized short-horizon GRPO improvements on an independent bridge set. Third, we train complete GRPO runs on selected subsets and evaluate final generative performance on MATH-500 and LiveCodeBench. This separation allows us to distinguish prediction of finite-step utility from its transfer to actual optimization and final downstream performance. Across six model-domain settings covering Qwen, DeepSeek, and Mistral on mathematics and code, DataLens achieves the highest point-estimate correlation with measured finite-step utility. On Qwen-Math, DataLens and the signed curvature score have similar global rank correlation, but DataLens better identifies the high-utility tail within the measured audit and shows stronger correlation with early realized GRPO gains. More broadly, the relationship between the score and realized GRPO improvement is strongest at short horizons and gradually weakens as training moves away from the initial model state, which is consistent with the local nature of the approximation.

These candidate-level gains translate into more effective training subsets. Under a common nonzero-probe eligibility rule, DataLens outperforms first-order alignment and signed curvature across all 24 model-domain-budget settings, and achieves the highest mean performance when selecting 100 examples in all six model-domain settings. The strongest result occurs on Qwen-Math, where 100 examples selected by DataLens reach 84.0% MATH-500 accuracy across three training seeds, compared with 70.1% using first-order alignment and 67.5% using random selection. Additional experiments show that this result remains stable across training, rollout, and candidate-pool randomness, while selection adds approximately 43%–47% of the accelerator cost of one downstream training run. Our results support a simple conclusion: local gradient signals are useful but incomplete proxies for RL data utility. Modeling how a target objective changes along candidate-induced finite update directions provides a more informative criterion for selecting training data under a limited optimization budget. Our main contributions are:

- We formulate finite-step data utility prediction and define a controlled target-objective improvement induced by a candidate-specific finite update.
- We introduce DataLens, which augments first-order target alignment with directional objective geometry and can be computed efficiently without constructing a full Hessian.
- We establish local error and ranking guarantees and validate DataLens through controlled probes, short-horizon GRPO, and data-efficient training across models and domains.

2 Related Work

Data selection and influence estimation for language models. Training-data selection has long been closely connected to influence estimation, gradient matching, and coreset construction (Mirzasoleiman et al., 2020; Killamsetty et al., 2021b;a; Paul et al., 2021). Influence Functions estimate the effect of infinitesimally upweighting an individual training example (Koh & Liang, 2017), while TracIn approximates training-example influence by accumulating gradient similarity along the optimization trajectory (Pruthi et al., 2020). TRAK scales data attribution to modern differentiable models through randomized projections (Park et al., 2023). At LLM scale, LESS extends gradient-based influence estimation to target-aware instruction-tuning data selection using low-dimensional, optimizer-aware representations (Xia et al., 2024), and IProX studies smaller proxy models that preserve influence information to improve the scalability of gradient-based selection (Chen et al., 2026). These works establish local gradient information as a practical proxy for downstream example value. In contrast, DataLens focuses on the target-objective change induced after a finite candidate-specific parameter update.

Data efficiency in LLM reinforcement learning. To boost data efficiency, modern reasoning-RL pipelines build on PPO-style objectives (Schulman et al., 2017), with GRPO originally introduced in DeepSeekMath (Shao et al., 2024) and subsequently used in large-scale reasoning RL (DeepSeek-AI, 2025). One-Shot RLVR shows that substantial reasoning gains can arise from small training sets (Wang et al., 2025); difficulty-based online selection combines informative success rates with rollout replay (Sun et al., 2025); and DEPO jointly considers offline filtering and online explorability (Tang et al., 2026). LearnAlign and GradAlign use gradient alignment to identify examples that are more likely to facilitate RLVR optimization (Li et al., 2026; Yang et al., 2026a). CROPI and InfOES further explore influence-based criteria for experience selection in RLVR or RLHF (Zhu et al., 2026; Gong et al., 2026). Second-Order Rollout improves data utilization by jointly exploiting response generation and critique generation (Yang et al., 2026b). DataLens addresses a complementary question: how the target objective evolves following the finite parameter update induced by a candidate. In our experiments, LearnAlign and GradAlign are instantiated as selection-time scoring functions at a common initial model state, aligning this comparison with our one-shot selection setting.

Optimization geometry and curvature. Higher-order geometric information has also been used to improve optimization itself. TayPO derives policy-optimization objectives from Taylor expansions (Tang et al., 2020). More recently, CGPO uses curvature-aware preconditioning to coordinate parameter updates across reasoning domains (Liang et al., 2026), while CAPO uses second-order policy-gradient geometry to improve optimization stability (Melo et al., 2026). These methods use geometry to modify the optimization process. By contrast, DataLens leverages directional geometry prior to training, ranking candidates according to the target-objective change along their induced update directions and then training the selected subset with standard GRPO.

3 Method

3.1 Problem Formulation

To quantify finite-step data utility, we first define a shared target objective. As shown in the "Validation samples" branch of Figure 2, a held-out target set $\mathcal{D}_v$ defines the objective used to assess every candidate. Let $\theta \in \mathbb{R}^d$ denote the model parameters. We use length-normalized teacher-forced log-likelihood,

$$J_v(\theta) = \frac{1}{|\mathcal{D}_v|} \sum_{(x,a)\in\mathcal{D}_v} \frac{1}{|a|} \sum_t \log \pi_\theta(a_t \mid x, a_{<t}), \tag{1}$$

where larger values indicate better target performance. For a prescribed candidate displacement s_i, controlled finite-step utility is

$$U_{\text{probe}}(i) = J_v(\theta + s_i) - J_v(\theta). \tag{2}$$

This is the "Prediction target" shown at the bottom of Figure 2. Our goal is to predict this change without separately executing every update and reevaluating J_v.

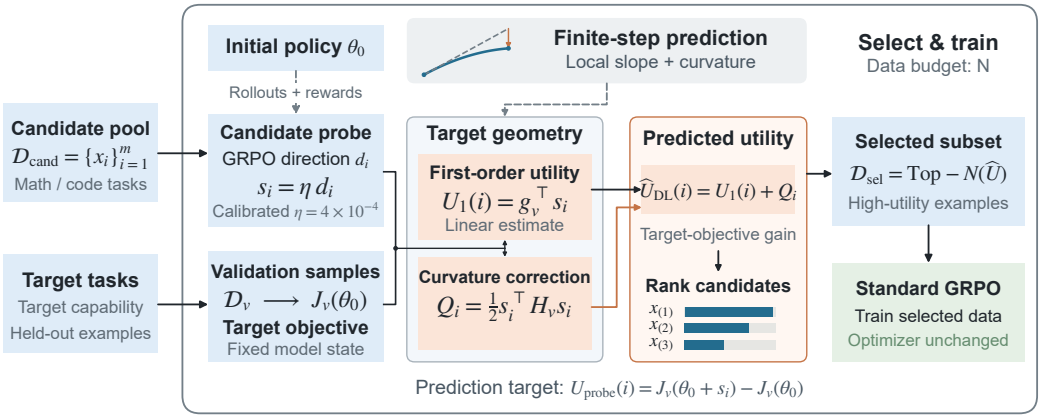

Figure 2: **DataLens framework.** Candidate probes and held-out validation samples jointly determine first-order utility and a directional curvature correction. Their sum predicts the target-objective gain of each finite displacement, yielding a ranking from which the Top-N candidates are selected for standard GRPO. The curve and rank bars are schematic.

To render candidate-level scoring practical, we restrict displacements to the LM head. Let $P_h \in \mathbb{R}^{d_h \times d}$ be the row-orthonormal projection onto this subspace. Candidate-level experiments measure

$$U^{(h)}_{\text{probe}}(i) = J_v\left(\theta_0 + P_h^\top s_i^{(h)}\right) - J_v(\theta_0). \tag{3}$$

Each candidate is evaluated independently at the same initial state θ_0, so its predicted gain can be compared with those of other candidates before subset selection.

3.2 Framework Overview

To predict this utility before training, we combine candidate updates with target-objective geometry. Figure 2 organizes DataLens into two input branches followed by scoring and selection. The "Candidate pool" feeds the "Candidate probe" block, while "Target tasks" supply the held-out validation objective. These branches meet in "Target geometry", where the candidate displacement and target objective jointly determine the first-order utility and curvature correction. Their sum produces the "Predicted utility" used by the "Select & train" stage.

Candidate-induced update. As shown in the "Initial policy" and "Candidate probe" blocks of Figure 2, all candidates are scored from the same policy π_{θ_0}. For prompt x_i, we sample $K = 8$ completions y_{ik}, obtain binary verifier rewards r_{ik}, and normalize rewards within the group. Writing $\bar{r}_i = K^{-1} \sum_k r_{ik}$, the group-relative advantage is

$$A_{ik} = \frac{r_{ik} - \bar{r}_i}{\sqrt{K^{-1} \sum_{k'=1}^{K} (r_{ik'} - \bar{r}_i)^2 + \epsilon_A}}, \qquad \epsilon_A = 10^{-6}. \tag{4}$$

At initialization, the GRPO policy ratio equals one, clipping remains inactive, and the KL term exhibits a zero first derivative (Shao et al., 2024). The candidate-induced projected direction is therefore

$$d_i^{(h)} = P_h \frac{1}{K} \sum_{k=1}^{K} \frac{A_{ik}}{|y_{ik}|} \sum_t \nabla_\theta \log \pi_{\theta_0}(y_{ikt} \mid x_i, y_{ik,<t}), \tag{5}$$

where P_h projects onto the LM head. This direction retains both the orientation and magnitude of the candidate's local training signal. Given the calibrated scale η, its finite displacement is

$$s_i^{(h)} = \eta d_i^{(h)}. \tag{6}$$

If all eight rewards agree, the advantages and direction vanish. All primary selectors, including Random, therefore rank the same eligible set

$$\mathcal{E} = \{i : \|d_i^{(h)}\|_2 > 0\}. \tag{7}$$

Zero-direction examples remain in the original 5,000-example accounting and the full-data reference.

Update-scale calibration. The calibrated scale in the "Candidate probe" block relates the controlled displacement to actual training. We match the median projected direction norms to the displacement of one real GRPO step on a disjoint 128-example Qwen-Math set. This yields $\eta_{\text{raw}} = 3.995 \times 10^{-4}$, rounded once to $\eta = 4 \times 10^{-4}$ for all main experiments. Appendix A.8 provides the protocol and recorded norms; benchmark outcomes and measured candidate utilities are not used in calibration.

3.3 Controlled-Probe Utility Prediction

With candidate displacements defined, we now approximate their effect on the target objective. As shown in the "Finite-step prediction" inset of Figure 2, the local slope gives a tangent approximation, while curvature accounts for the objective's departure from that tangent along a finite displacement. Let $g_v = \nabla J_v(\theta_0)$ and $H_v = \nabla^2 J_v(\theta_0)$. For a smooth objective,

$$J_v(\theta_0 + s_i) - J_v(\theta_0) = g_v^\top s_i + \tfrac{1}{2} s_i^\top H_v s_i + R_3(s_i). \tag{8}$$

Adding the "First-order utility" and "Curvature correction" terms gives the "Predicted utility" score,

$$\widehat{U}_{\text{DL}}(i) = g_v^\top s_i + \tfrac{1}{2} s_i^\top H_v s_i = \eta g_v^\top d_i + \tfrac{\eta^2}{2} d_i^\top H_v d_i. \tag{9}$$

The first term quantifies initial alignment with target improvement; the second corrects this prediction for the objective's curvature along the finite update. The shared incoming branch to "Target geometry" in Figure 2 emphasizes that both terms depend on the candidate displacement and the held-out objective. To distinguish these signals in the experiments, we define

$$U_1(i) = \eta g_v^\top d_i, \qquad q_i = d_i^\top H_v d_i, \qquad q_{\text{dir},i} = \frac{d_i^\top H_v d_i}{\|d_i\|_2^2}. \tag{10}$$

The correction labeled Q_i in Figure 2 is therefore $\eta^2 q_i/2$. Raw q retains update magnitude, whereas q_{dir} isolates normalized directional curvature. For the detached LM-head objective, H_v is negative semidefinite, so $q_i \le 0$: larger curvature scores correspond to a weaker local reduction in the target objective.

Projected implementation. To make this prediction tractable for LLMs, we implement the "Target geometry" block in the LM-head subspace with the model body detached. Figure 2 omits the projection superscripts for readability; the implemented score is

$$\widehat{U}_{\text{DL}}^{(h)}(i) = (g_v^{(h)})^\top s_i^{(h)} + \tfrac{1}{2} (s_i^{(h)})^\top H_v^{(h)} s_i^{(h)}, \tag{11}$$

with $g_v^{(h)} = P_h g_v$ and $H_v^{(h)} = P_h H_v P_h^\top$. Its directional curvature admits an analytic predictive-Fisher/Gauss–Newton form (Martens, 2020), avoiding explicit construction of the Hessian. Appendix A.9 derives this form and the finite-difference extension for richer subspaces; Appendix A.10 audits the implementation. As shown in the "Rank candidates" and "Selected subset" blocks of Figure 2, we rank eligible candidates by predicted utility and retain the Top-N examples under the data budget. The selected subset then enters the "Standard GRPO" block with the downstream optimizer unchanged. The bridge and downstream experiments test how this ranking transfers to jointly trained subsets.

3.4 Theoretical Guarantees and Scope

To establish when the quadratic score yields reliable predictions and rankings, we first bound its approximation error. Define $\phi(t) = J_v(\theta_0 + ts)$, so the quadratic score is $\widehat{U}_{\text{DL}}(s) = \phi'(0) + \frac{1}{2}\phi''(0)$. Assume J_v is three-times continuously differentiable along the segment and $|\phi'''(t)| \le M\|s\|^3$ for all $t \in [0,1]$. Taylor's integral remainder and this derivative bound give

$$\begin{aligned} U_{\text{probe}}(s) - \widehat{U}_{\text{DL}}(s) &= \tfrac{1}{2} \int_0^1 (1-t)^2 \phi'''(t)\, dt, \\ |U_{\text{probe}}(s) - \widehat{U}_{\text{DL}}(s)| &\le \tfrac{M}{2}\|s\|^3 \int_0^1 (1-t)^2\, dt = \tfrac{M}{6}\|s\|^3. \end{aligned} \tag{12}$$

This bound also gives a sufficient condition for preserving candidate order. If the predicted utility gap satisfies

$$\left|\widehat{U}_{\text{DL}}(i) - \widehat{U}_{\text{DL}}(j)\right| > \frac{M}{6}\left(\|s_i\|^3 + \|s_j\|^3\right), \tag{13}$$

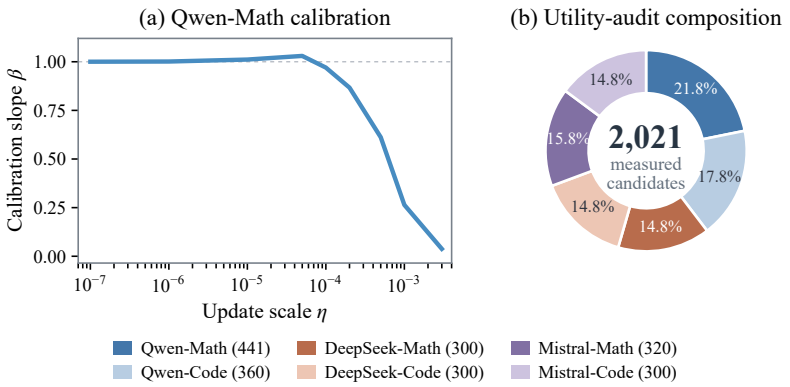

Figure 3: **Calibration and audit coverage.** (a) Qwen-Math calibration slopes from Table 14; the dashed line marks unit slope. (b) Shares of the 2,021 measured candidates across six settings in Table 1. Legend entries give counts; percentages are rounded.

then the ordering of candidates i, j agrees with their exact controlled utility. Thus, sufficiently separated scores are robust to the two higher-order remainders. Appendix B states the smoothness assumptions and supplies both proofs.

These guarantees pertain to local marginal utility. Joint subset updates contain cross-candidate terms (Eq. (24)), and real GRPO introduces resampling, optimizer state, clipping, and parameter drift. The theory therefore motivates candidate screening; transfer to jointly trained subsets requires the empirical tests presented below.

4 Experiments

4.1 Experimental Setup

To connect local prediction with practical data selection, we evaluate three progressively more realistic outcomes: exact controlled utility, short-horizon GRPO improvement, and final performance after subset training. The **Qwen, DeepSeek, and Mistral** model families are tested on **math** and **code**. Math candidates come from cleaned OpenR1-Math-220k and are evaluated on MATH-500; code candidates come from TACO and are evaluated on LiveCodeBench release-v6. Candidate, scoring-validation, bridge, calibration, and benchmark prompts are mutually disjoint following exact and near-duplicate clustering. Benchmark prompts serve only for decontamination and final evaluation; their outcomes never enter scoring, calibration, ranking, updates, or early stopping. Appendix A records dataset sizes, splits, verifiers, and training settings.

Within each pool seed, all rollout-dependent selectors receive the same cached completions and rewards and rank the common set $\mathcal{E}$. At each data budget, matched optimization settings isolate the effect of subset choice. Scoring follows the calibrated probe protocol, and numerical audits use a perturbation radius checked against the analytic reference. Table 8 records the shared training configuration; Appendix A gives calibration and validation details. Paired-bootstrap confidence intervals resample candidates for utility audits and minibatch groups for the bridge.

4.2 Controlled and Numerical Validation

Before testing candidate rankings, we check whether directional geometry explains the error left by the first-order estimate. Define $Q_i = \frac{1}{2} s_i^\top H_v s_i$ and $R_i = U_{\text{probe}}(i) - g_v^\top s_i$. Equation (8) gives $R_i = Q_i + R_3(s_i)$, so their calibration slope should approach one locally. Figure 3(a) shows that the curvature correction tracks the first-order residual in the local regime, with the same pattern across settings in Table 15. This connects the measured residual to the directional term used for selection, giving the correction a direct objective-level interpretation. In the LM-head setting, the

Table 1: **Candidate-level projected controlled-probe utility prediction.** Spearman correlation with exactly measured $U^{(h)}_{\text{probe}}$. Counts are fixed-hash uniform samples from the predefined nonzero-direction stratum, sampled before measuring utility. Best point estimates are bold.

	Mathematical Reasoning			Code Generation		
Method	Qwen	DeepSeek	Mistral	Qwen	DeepSeek	Mistral
Measured candidates (n)	441	300	320	360	300	300
U_1	.065	.084	.071	.098	.083	.072
Success rate p	.302	.331	.307	.325	.301	.283
Difficulty $p(1-p)$	.442	.431	.399	.425	.404	.381
Gradient norm $\|d\|$	.514	.472	.438	.459	.433	.401
Cosine alignment	.143	.178	.161	.196	.181	.169
LearnAlign (one-shot score)	.544	.552	.509	.526	.501	.470
GradAlign (one-shot score)	.552	.560	.516	.534	.508	.477
Empirical-Fisher finite-step	.561	.567	.522	.539	.512	.481
q	.639	.548	.503	.526	.493	.462
q_{dir}	.602	.526	.481	.505	.472	.441
DataLens	**.660**	**.587**	**.536**	**.558**	**.528**	**.497**

signed correction discounts directions with a sharper predicted decline in marginal gain. Among similarly aligned candidates, this favors updates predicted to retain more of their initial benefit over the prescribed displacement. Calibration weakens at larger displacements as higher-order terms become more prominent, motivating separate ranking and scale-sensitivity tests at the training-calibrated update scale. The fitted error orders also support the expected improvement from a linear to a quadratic approximation (Appendix B).

Independent direction and finite-difference audits check the numerical consistency of the scoring pipeline before we evaluate ranking. The direction comparison tests whether the implemented probe follows the intended policy-gradient update, while the curvature comparison checks agreement between analytic and numerical estimates (Appendix A.10). Relative error and tail overlap complement correlation here: the audit radius should preserve useful candidates as well as the broad ordering. The measured ranking reversal in Table 16 illustrates the resulting benefit. An initially favorable direction can lose more of its predicted gain over a finite displacement, and the curvature correction helps distinguish it from a candidate with a stronger realized improvement.

4.3 Candidate-Level Utility Prediction

Having checked the scoring pipeline, we next test whether its scores rank candidates by exact controlled utility. For each model-domain setting, a fixed-hash uniform sample is drawn from the predefined nonzero-direction stratum before measuring any true utility. The exact projected controlled finite-step utility $U^{(h)}_{\text{probe}}$ provides the ranking target. As shown in Table 1, DataLens has the highest Spearman point estimate across settings, and paired-bootstrap intervals support its advantage over U_1 throughout. Directional geometry thus improves candidate ordering beyond local alignment. The weaker success-rate and difficulty rankings also indicate that how often a prompt is solved does not alone determine the value of its update. The combined score evaluates both initial target alignment and how much of that gain survives the prescribed displacement.

Selection depends most directly on the high-utility tail, where similar global correlations can conceal practically relevant differences. Within the measured Qwen-Math audit, DataLens retrieves more high-utility candidates and yields greater mean utility for the selected subset than q (Table 17). The candidates selected only by DataLens also have higher mean measured utility than those selected only by q, locating the benefit in the examples exchanged between the two subsets. Thus, the improved overlap accompanies better retained utility, linking ranking changes to the quality of the selected data. Better audit-tail recovery also appears across the other settings (Table 19). The tail gains support DataLens on Qwen-Math despite an inconclusive global-ranking comparison with q (Appendix C.3). This evidence concerns the measured audits, rather than recovery of the full-pool optimum.

4.4 BRIDGING TO SHORT-HORIZON REAL GRPO

We next move from controlled candidate updates to minibatch GRPO training to test whether the ranking signal transfers. Sampled eligible candidates are partitioned into disjoint minibatch-sized groups G. For candidate score $S_m(i)$, the group score is

$$S_m(G) = \frac{1}{|G|} \sum_{i \in G} S_m(i). \tag{14}$$

This matches the granularity of a real minibatch by averaging marginal scores, without explicit cross-candidate interaction terms. Every group starts from θ_0 and a fresh AdamW state. Its prompts are reused over a short GRPO trajectory with fresh completions at each step. Scores use $\mathcal{D}_v$, whereas the realized gain after r steps is measured independently:

$$\Delta J_{\text{bridge}}(r) = J_{\text{bridge}}(\theta_r) - J_{\text{bridge}}(\theta_0). \tag{15}$$

This separation tests transfer to real optimization rather than remeasuring the score-defining objective. Appendix D.7 records the full protocol and Table 27 reports setting-level results.

Table 2: **Bridge to real GRPO.** Mean Spearman correlation with bridge-set gain after r steps.

Horizon	U_1	q	q_{dir}	DataLens
1 step	.16	.43	.40	**.53**
5 steps	.14	.40	.37	**.50**
10 steps	.11	.35	.33	**.45**
20 steps	.08	.28	.26	**.38**

As shown in Table 2, DataLens maintains the highest cross-setting mean correlation at every evaluated horizon, including the longest trajectory. Although all initialization-based scores become less predictive as training proceeds, DataLens continues to provide a useful ranking signal through successive GRPO updates with fresh rollouts and evolving optimizer and parameter states. Its mean correlation margin over raw curvature remains approximately stable across these horizons, suggesting that the combined geometric signal retains relative value as training progresses. The setting-level point estimates show the same pattern (Table 27). On Qwen-Math, paired intervals support its advantage over q at earlier horizons; the longest-horizon estimate remains higher, although its interval includes zero. Independent bridge-set evaluation supports transfer of this persistent signal to subsequent learning and complements the end-to-end evidence below.

4.5 END-TO-END DATA EFFICIENCY

The short-horizon results motivate a direct test of data efficiency through complete subset-training runs. We choose each selector's Top-N examples from the common eligible set under the same GRPO configuration and optimization budget. As shown in Table 3, DataLens consistently improves on first-order alignment and raw curvature across the model–domain–budget grid, with broad gains over normalized curvature. The matched-budget comparison with stronger one-shot selectors gives the same overall pattern: DataLens has the highest point estimate across settings, with the clearest separation on Qwen-Math (Table 22). The shared training recipe attributes these gains to allocating the optimization budget toward candidates ranked by finite-step utility.

The candidate-level and downstream results also distinguish raw curvature q from normalized curvature q_{dir}. Raw curvature better matches the controlled probe, whereas normalized curvature yields stronger downstream means across the budget grid. Initial update magnitude is informative for a matched displacement, while normalization compares directional curvature per unit update (Appendix D.2). DataLens combines alignment and curvature in a single utility estimate, favoring candidate directions whose initial gains remain favorable after the finite-step correction. Its downstream gains over either component support this combined criterion for selecting data that will subsequently be trained together.

Exposure-matched and tail-only controls indicate that the Qwen-Math peak depends on both subset composition and repeated exposure to selected examples (Table 26). Increasing subset size under a fixed training budget spreads updates over more examples and introduces a lower-ranked tail. Matching exposure recovers part of the lost performance, while training on the tail alone remains weaker. Duplicating the selected subset largely preserves performance, and score-proportional sampling improves the larger subset over uniform sampling. These observations are consistent with the ranking identifying productive destinations for repeated training draws, suggesting value for allocating exposure as well as choosing subset membership. A compact subset therefore benefits from

Table 3: **End-to-end RL performance under different training-data budgets.** Entries are mean MATH-500 exact-match accuracy or LiveCodeBench Pass@1 over three training seeds. All selector rows, including Random, use the common nonzero-direction eligibility stratum. All cells use 8 prompts/update, 8 rollouts/prompt, and 200 optimizer steps.

	Qwen				**DeepSeek**				**Mistral**			
	Selected examples (N)				Selected examples (N)				Selected examples (N)			
Method	25	50	100	200	25	50	100	200	25	50	100	200
Mathematical Reasoning												
U_1	.644	.673	.701	.719	.633	.661	.686	.705	.604	.631	.657	.676
q	.660	.688	.714	.731	.648	.676	.701	.718	.618	.645	.671	.689
q_{dir}	.667	.695	.721	.738	.654	.682	.707	**.738**	.624	.651	.677	.695
DataLens	**.683**	**.711**	**.840**	**.751**	**.669**	**.696**	**.720**	.736	**.637**	**.665**	**.690**	**.707**
Random	.620	.648	.675	.688	.611	.639	.663	.681	.583	.611	.635	.653
Code Generation												
U_1	.371	.401	.428	.447	.349	.379	.404	.423	.332	.360	.386	.406
q	.383	.412	.440	.457	.359	.389	.415	.433	.342	.370	.397	.416
q_{dir}	.389	.418	.446	.463	.364	.394	.421	.439	**.360**	.376	.403	.421
DataLens	**.401**	**.431**	**.458**	**.475**	**.376**	**.406**	**.433**	**.451**	.358	**.388**	**.415**	**.434**
Random	.352	.381	.405	.423	.331	.359	.383	.401	.314	.341	.366	.384

both concentrating useful examples and giving their updates sufficient training exposure. The selected subsets approach the full-data reference in most settings and exceed it on Qwen-Math (Table 23), illustrating how selection can make limited training effort more effective.

4.6 Robustness and Compute Trade-off

Beyond final performance, we examine whether the gains remain stable across sampling choices and justify the added scoring cost. On Qwen-Math, fixed-subset runs show limited sensitivity to the training seed, and performance remains in a similar range when candidate-pool, rollout/scoring, and training randomness vary jointly (Table 25). This stability despite changing selection membership suggests that useful subsets need not contain exactly the same examples. Reducing the target-set size also preserves DataLens's ranking advantage over raw curvature and retains strong downstream performance, while overlap with the default selection increases as the target set grows (Table 21). Nearby update scales similarly preserve much of the selected set and utility correlation (Appendix C.5). These controls support the stability of the selection signal across the tested sampling and scoring choices, beyond a single fixed subset or scale.

Selection adds a material upfront compute cost. Nevertheless, DataLens retains an advantage over random selection at approximately matched total accelerator cost on both Qwen domains (Appendix A.12). The relevant comparison is therefore how effectively the total budget is used: spending part of it on identifying useful candidates can outperform allocating the same resources to additional training on randomly selected data. The projected-subspace ablation further supports the LM-head implementation as a practical compromise, capturing much of the measured ranking signal at lower scoring cost than richer subspaces (Table 11).

5 Conclusion

These results support **DataLens**, which combines first-order target alignment with directional curvature to predict finite-step data utility in LLM reinforcement learning. Computing this geometry in the LM-head subspace makes candidate scoring practical without constructing the full Hessian. Across math and code tasks, controlled probes show improved utility ranking, and bridge experiments show the highest mean correlation with GRPO gains at every evaluated horizon. These signals translate into stronger training subsets under matched budgets, with stable performance across the tested sources of randomness and gains under compute-matched comparison. The results support allocating RL training effort according to predicted finite-update gains, connecting local objective geometry to practical data selection while leaving the downstream optimizer unchanged.

AI USE STATEMENT

We used AI tools to assist with literature searches.

REFERENCES

Sirui Chen, Yunzhe Qi, Mengting Ai, Yifan Sun, Ruizhong Qiu, Jiaru Zou, and Jingrui He. Influence-preserving proxies for gradient-based data selection in LLM fine-tuning. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=PDNpRLxDlI.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. URL https://arxiv.org/abs/2501.12948.

Yifan Gong, Jing Yao, Xiting Wang, Xunlong Wang, Xiaoyuan Yi, and Xing Xie. Influence-based online experience selection for effective RLHF. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 47755–47771, 2026. doi: 10.18653/v1/2026.acl-long.2206. URL https://aclanthology.org/2026.acl-long.2206/.

Krishnateja Killamsetty, Durga S, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. GRAD-MATCH: Gradient matching based data subset selection for efficient deep model training. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5464–5474. PMLR, 2021a. URL https://proceedings.mlr.press/v139/killamsetty21a.html.

Krishnateja Killamsetty, Durga Sivasubramanian, Ganesh Ramakrishnan, and Rishabh Iyer. GLISTER: Generalization based data subset selection for efficient and robust learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 8110–8118, 2021b. doi: 10.1609/aaai.v35i9.16988. URL https://ojs.aaai.org/index.php/AAAI/article/view/16988.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1885–1894, 2017. URL https://proceedings.mlr.press/v70/koh17a.html.

Shipeng Li, Zhiqin Yang, Shikun Li, Xiaobo Xia, Hengyu Liu, Xinghua Zhang, Gaode Chen, Dong Fang, Ying Tai, and Zhe Peng. LearnAlign: Data selection for LLM reinforcement learning with improved gradient alignment. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 40417–40434, 2026. doi: 10.18653/v1/2026.findings-acl.2009. URL https://aclanthology.org/2026.findings-acl.2009/.

Xize Liang, Lin Yang, Jie Wang, Rui Liu, Yang Lu, Jinliang Zeng, Hanzhu Chen, Dong Li, and Jianye Hao. Boosting multi-domain reasoning of LLMs via curvature-guided policy optimization. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/f0bb0cc7cd2a027cff5f237f28def7eb-Abstract-Conference.html.

James Martens. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21(146):1–76, 2020. URL https://jmlr.org/papers/v21/17-678.html.

Luckeciano C. Melo, Alessandro Abate, and Yarin Gal. Stabilizing policy gradients for sample-efficient reinforcement learning in LLM reasoning. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2510.00819.

Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 6950–6960. PMLR, 2020. URL https://proceedings.mlr.press/v119/mirzasoleiman20a.html.

Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. TRAK: Attributing model behavior at scale. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 27074–27113. PMLR, 2023. URL https://proceedings.mlr.press/v202/park23c.html.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. In *Advances in Neural Information Processing Systems*, volume 34, pp. 20596–20607, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/ac56f8fe9eea3e4a365f29f0f1957c55-Abstract.html.

Barak A. Pearlmutter. Fast exact multiplication by the hessian. *Neural Computation*, 6(1):147–160, 1994. doi: 10.1162/neco.1994.6.1.147. URL https://doi.org/10.1162/neco.1994.6.1.147.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/e6385d39ec9394f2f3a354d9d2b88eec-Abstract.html.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300.

Yifan Sun, Jingyan Shen, Yibin Wang, Tianyu Chen, Zhendong Wang, Mingyuan Zhou, and Huan Zhang. Improving data efficiency for LLM reinforcement fine-tuning through difficulty-targeted online data selection and rollout replay. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://papers.nips.cc/paper_files/paper/2025/hash/fd54731cc635e2ed6033e8a0e193eddf-Abstract-Conference.html.

Xinyu Tang, Zhenduo Zhang, Yurou Liu, Wayne Xin Zhao, Zujie Wen, Zhiqiang Zhang, and Jun Zhou. Towards high data efficiency in reinforcement learning with verifiable reward. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2509.01321.

Yunhao Tang, Michal Valko, and Remi Munos. Taylor expansion policy optimization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9397–9406, 2020. URL https://proceedings.mlr.press/v119/tang20d.html.

Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Liyuan Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, Weizhu Chen, Shuohang Wang, Simon S. Du, and Yelong Shen. Reinforcement learning for reasoning in large language models with one training example. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://papers.neurips.cc/paper_files/paper/2025/hash/b1ea3f93167c55f3fc2999b68a170ff3-Abstract-Conference.html.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting influential data for targeted instruction tuning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54104–54132, 2024. URL https://proceedings.mlr.press/v235/xia24c.html.

Ningyuan Yang, Weihua Du, Weiwei Sun, Sean Welleck, and Yiming Yang. GradAlign: Gradient-aligned data selection for LLM reinforcement learning. In *Conference on Language Modeling*, 2026a. URL https://arxiv.org/abs/2602.21492.

Zhe Yang, Yudong Wang, Rang Li, and Zhifang Sui. Towards better rl training data utilization via second-order rollout. *arXiv preprint arXiv:2602.22765*, 2026b. URL https://arxiv.org/abs/2602.22765.

Erle Zhu, Dazhi Jiang, Yuan Wang, Xujun Li, Jiale Cheng, Yuxian Gu, Yilin Niu, Aohan Zeng, Jie Tang, Minlie Huang, and Hongning Wang. Data efficient RLVR via off-policy influence guidance. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 46167–46192, 2026. doi: 10.18653/v1/2026.acl-long.2141. URL https://aclanthology.org/2026.acl-long.2141/.

A Experimental Details

A.1 Models and Software

Table 4: **Model checkpoints used in the experiments.**

Setting	Checkpoint
Math / Qwen	`Qwen/Qwen2.5-Math-7B`
Math / DeepSeek	`deepseek-ai/deepseek-math-7b-base`
Math / Mistral	`mistralai/Mathstral-7B-v0.1`
Code / Qwen	`Qwen/Qwen2.5-Coder-7B`
Code / DeepSeek	`deepseek-ai/deepseek-coder-6.7b-base`
Code / Mistral	`mistralai/Mistral-7B-v0.3`

A.2 Data Partitions and Leakage Control

The Math reservoir is `open-r1/OpenR1-Math-220k`, configuration `default`, train split. The Code reservoir is `BAAI/TACO`, configuration `ALL`, train split. The fixed sampling seed is 20250214.

Table 5: **Data roles and sizes.**

Domain	Role	Dataset/split	n
Math	Source reservoir	OpenR1-Math-220k/default/train	93,733 upstream rows
Math	$\mathcal{D}_{\text{cand}}$	cluster-disjoint source sample	5,000
Math	$\mathcal{D}_v$	held-out source clusters	256
Math	$\mathcal{D}_{\text{bridge}}$	independent source clusters	256
Math	$\mathcal{D}_{\text{cal}}$	independent source clusters	128
Math	$\mathcal{D}_{\text{test}}$	MATH-500	500
Code	Source reservoir	TACO/ALL/train	25,443 raw
Code	$\mathcal{D}_{\text{cand}}$	cluster-disjoint source sample	5,000
Code	$\mathcal{D}_v$	held-out source clusters	256
Code	$\mathcal{D}_{\text{bridge}}$	independent source clusters	256
Code	$\mathcal{D}_{\text{cal}}$	independent source clusters	128
Code	$\mathcal{D}_{\text{test}}$	LiveCodeBench release-v6 window	454

We canonicalize Unicode, whitespace, LaTeX, and numeric formats; hash canonical prompts with SHA-256; and cluster token 5-gram MinHash matches at Jaccard similarity at least .85. Math additionally normalizes variable names and option labels; Code normalizes statements and I/O specifications. Every near-duplicate cluster is assigned wholly to one split, and pairs in the borderline interval $[.80, .85)$ are manually reviewed.

Table 6: **Decontamination accounting against the benchmark prompts.**

Domain	Valid/source-deduped	Exact removed	Near-dup. removed	Clean reservoir
Math	88,450	117	84	88,249
Code	17,834	23	41	17,770

Benchmark prompts are used for decontamination and post-training reporting. Benchmark outcomes do not enter the scoring or training pipeline.

A.3 Candidate Flow and the Common Eligibility Rule

All 5,000 source candidates are processed by the shared eight-rollout probe. With binary group-relative rewards, an all-identical reward group has $A_{ik} = 0$ and hence $d_i^{(h)} = 0$. Such examples are retained in the source accounting and full-data reference but are excluded from every primary one-shot selector comparison through the common eligibility rule $\mathcal{E} = \{i : \|d_i^{(h)}\|_2 > 0\}$. This rule is applied before inspecting any measured utility or benchmark result.

Table 7: **Candidate flow.** Measured-U_{probe} samples are fixed-hash samples from the common nonzero-probe stratum.

Setting	Source pool	Zero direction	Eligible nonzero	Measured U_{probe}
Qwen-Math	5,000	1,738 (34.8%)	3,262	441
DeepSeek-Math	5,000	2,094 (41.9%)	2,906	300
Mistral-Math	5,000	2,276 (45.5%)	2,724	320
Qwen-Code	5,000	2,581 (51.6%)	2,419	360
DeepSeek-Code	5,000	2,764 (55.3%)	2,236	300
Mistral-Code	5,000	2,913 (58.3%)	2,087	300

Because the largest budget is $N = 200$, all six settings have ample eligible candidates. The common mask is also applied to the Random row of the primary selector tables, so those rows are eligibility-matched budget references.

A.4 Final Evaluation and Verifiers

MATH-500 uses all 500 problems, greedy decoding ($T = 0$), a maximum of 2,048 completion tokens, and normalized exact-match accuracy. Code uses the 454 problems in the LiveCodeBench `code_generation_lite` release-v6 window from 2024-08-01 through 2025-05-01, with the same greedy decoding and maximum completion length.

The Math verifier performs boxed-answer extraction, LaTeX normalization, and symbolic equivalence. The Code verifier runs TACO public tests and assigns reward one only when all tests pass. Each program has a 10-second, 1-GiB, network-disabled sandbox limit; 128 CPU workers execute the verification queue.

For J_v and J_{bridge}, Math uses one canonical verified solution per prompt. Code deterministically takes the first verifier-passing reference program after validity filtering. This single-reference construction creates a surrogate mismatch between selection-time teacher-forced objectives and final generative evaluation.

A.5 Downstream RL Configuration

Table 8: **Shared downstream GRPO configuration.**

Hyperparameter	Value	Hyperparameter	Value
Optimizer	AdamW	Global prompt batch	8
Learning rate	4×10^{-6}	Micro prompt batch/GPU	1
Weight decay	0.01	Gradient accumulation	1
Adam betas	$(0.9, 0.999)$	Generations/prompt K	8
Adam ϵ	10^{-8}	Completions/update	64
Scheduler	Cosine	Optimizer steps	200
Warmup ratio/steps	$0.05/10$	Total prompt draws	1,600
Gradient clip norm	1.0	Sampled completions	12,800
GRPO clip ϵ_c	0.2	KL coefficient β	0.04
Advantage ϵ_A	10^{-6}	Reference policy	Frozen π_{θ_0}
Rollout temperature	0.7	Top-p	0.95
Math prompt/completion max	$1024/2048$	Code prompt/completion max	$1536/2048$
Model precision	BF16	Optimizer state	FP32
Distributed training	FSDP FULL_SHARD	Rollout engine	vLLM
Gradient checkpointing	Enabled	Primary seeds	42, 7, 1337

A.6 Candidate Update Construction

All candidates are scored from the same initial model state θ_0. For a candidate prompt x_i, we sample $K = 8$ completions from the initial policy,

$$y_{ik} \sim \pi_{\theta_0}(\cdot \mid x_i),$$

and evaluate them with a binary verifier,

$$r_{ik} \in \{0, 1\}.$$

These rollouts and rewards characterize the local training signal available from candidate i. To match the group-relative learning signal used by downstream GRPO, we normalize rewards within each candidate group. Let

$$\bar{r}_i = \frac{1}{K} \sum_{k=1}^{K} r_{ik},$$

and use the normalized advantage defined in Eq. (4).

At the initialization point θ_0, the GRPO policy ratio is one, clipping is inactive, and the KL term has zero first derivative (Shao et al., 2024). The local GRPO update direction induced by candidate i is therefore given by Eq. (5), where P_h projects the gradient onto the LM-head parameter subspace defined above.

The direction $d_i^{(h)}$ represents the local training direction induced by candidate i at the common initial model state. Different candidates can vary both in direction and magnitude, and both factors can affect how the target objective changes under a finite update. Given an update scale η, we define the corresponding candidate-specific displacement as

$$s_i^{(h)} = \eta d_i^{(h)}. \tag{16}$$

The finite-step utility introduced in the previous section,

$$U_{\text{probe}}^{(h)}(i) = J_v\Big(\theta_0 + P_h^\top s_i^{(h)}\Big) - J_v(\theta_0),$$

therefore measures the actual change in the target objective after moving a finite distance along the GRPO direction induced by candidate i.

This construction also determines which candidates provide a valid update signal at scoring time. With binary rewards, if all $K = 8$ completions of a candidate receive the same reward, then all group-relative advantages are zero and hence

$$d_i^{(h)} = 0.$$

Such candidates do not produce a nonzero local update direction under the current one-shot probe. To ensure that differences between selectors arise from their scoring criteria, all primary selector comparisons, including Random, are performed on the common nonzero-direction set

$$\mathcal{E} = \left\{ i : \|d_i^{(h)}\|_2 > 0 \right\}. \tag{17}$$

Zero-direction examples remain part of the original 5,000-example accounting and the full-data reference.

A.7 One-Shot Baseline Protocol

All rollout-dependent selectors use the same cached $K = 8$ completions and rewards and the same eligibility stratum. Success rate ranks by $p_i = K^{-1} \sum_k r_{ik}$. Difficulty uses $p_i(1 - p_i)$. Gradient norm uses $\|d_i\|_2$, and cosine alignment uses

$$\frac{g_v^\top d_i}{\|g_v\|_2 \|d_i\|_2}.$$

LearnAlign and GradAlign are evaluated only through their static pre-training scoring components at θ_0, with the shared cached rollouts and target set; no adaptive rescoring is performed during the subsequent 200-step run. Accordingly, all tables label them "one-shot score" rather than claiming a comparison against their full adaptive procedures.

For the empirical-Fisher baseline,

$$F_{\text{emp}} = \frac{1}{|\mathcal{D}_v|} \sum_{(x,a) \in \mathcal{D}_v} \frac{1}{|a|} \sum_t z_t z_t^\top, \qquad z_t = P_h \nabla_\theta \log \pi_{\theta_0}(a_t \mid x, a_{<t}), \tag{18}$$

and

$$S_{\text{EF}}(i) = \eta g_v^\top d_i - \frac{\eta^2}{2} d_i^\top F_{\text{emp}} d_i. \tag{19}$$

This differs from DataLens because F_{emp} uses observed-target score outer products, whereas Eq. (21) uses the predictive softmax covariance, which is the exact detached-head Hessian.

A.8 Probe-Scale Calibration

The 128-example Qwen-Math calibration set is partitioned into sixteen non-overlapping eight-prompt minibatches. For each minibatch, both policy and optimizer state are reset to θ_0, eight rollouts are generated per prompt, and one GRPO optimizer step is taken. The projected displacement is

$$\Delta_b^{(h)} = P_h(\theta_b^+ - \theta_0).$$

The recorded medians are

$$m_\Delta = \text{median}_b \, \|\Delta_b^{(h)}\|_2 = 1.884 \times 10^{-2},$$

and

$$m_d = \text{median}_i \, \|P_h d_i\|_2 = 47.16.$$

Hence

$$\eta_{\text{raw}} = \frac{m_\Delta}{m_d} = 3.995 \times 10^{-4},$$

rounded once to $\eta = 4 \times 10^{-4}$. No measured U_{probe} value, downstream result, or benchmark outcome is used in this calibration.

A.9 Projected LM-head Geometry and Finite-Difference Extension

Let

$$s_i^{(h)} = P_h s_i, \qquad g_v^{(h)} = P_h g_v, \qquad H_v^{(h)} = P_h H_v P_h^\top.$$

The implemented score is Eq. (11). For one teacher-forced target token with detached hidden state h, target token y, LM-head weight matrix W, and predictive distribution $p = \mathrm{softmax}(Wh)$,

$$\nabla_W^2 \log p(y \mid h) = -\big(\mathrm{diag}(p) - pp^\top\big) \otimes hh^\top. \tag{20}$$

For an LM-head direction D, define $a = Dh$. The corresponding directional second derivative is

$$D^2 \log p(y \mid h)[D, D] = -a^\top(\mathrm{diag}(p) - pp^\top)a = -\operatorname{Var}_{k\sim p}(a_k). \tag{21}$$

Because J_v is the length-normalized average of these token log-likelihoods, for the matrix-shaped direction D_i corresponding to candidate i,

$$q_i = -\frac{1}{|\mathcal{D}_v|} \sum_{(x,a)\in\mathcal{D}_v} \frac{1}{|a|} \sum_t \operatorname{Var}_{k\sim p_t}((D_i h_t)_k)\,. \tag{22}$$

Thus $d_i^\top H_v d_i$ is available directly as a scalar without constructing the Hessian. This is the predictive-Fisher/Gauss–Newton geometry of the detached LM head and differs from an empirical Fisher based on observed-target score outer products.

Hessian-vector products can be evaluated without explicitly materializing the full Hessian (Pearlmutter, 1994). For richer differentiated subspaces, where the analytic detached-head structure no longer applies, we use the centered finite-difference Hessian-vector approximation

$$H_v d_i \approx \frac{\nabla J_v(\theta_0 + \epsilon d_i) - \nabla J_v(\theta_0 - \epsilon d_i)}{2\epsilon}, \tag{23}$$

with directional curvature $d_i^\top(H_v d_i)$. The perturbation radius is $\epsilon\|d_i\|_2$, so ϵ trades numerical precision against truncation error. We choose the audit radius by comparison with the analytic LM-head reference, evaluating both score agreement and preservation of the high-utility tail on an independent candidate sample. Table 10 reports the finite-difference sensitivity across radii.

A.10 Update-Direction and Curvature Validation

Table 9: **Update-direction sanity check over 50 Qwen-Math candidates.** Cosine summaries are reported to two decimal places.

Metric	Result
Mean cosine similarity	.97
Minimum cosine similarity	.96
Maximum cosine similarity	1.00
Candidates with negative cosine	0/50

Table 10: **Exact/direct curvature versus finite differences over 200 Qwen-Math candidates.** The operational radius (marked "used") balances score agreement with low relative error and high tail overlap. Bold marks the best value in each metric column.

ϵ	Spearman	Pearson	Median rel. error	Top-25	Top-50	Top-100
10^{-4}	.970	.970	4.1%	23/25	47/50	96/100
3×10^{-4}	.969	.969	2.5%	24/25	48/50	97/100
10^{-3} (used)	.975	.975	**1.6%**	25/25	49/50	98/100
3×10^{-3}	.970	.970	2.3%	24/25	48/50	97/100
10^{-2}	**.983**	**.987**	6.4%	22/25	45/50	92/100

A.11 PROJECTED-SUBSPACE ABLATION

Table 11: **Differentiated-subspace ablation on Qwen-Math.**

Differentiated subspace	Utility Spearman	Top-100 overlap	Rank agreement	Rel. cost
LM head	.660	100%	1.000	1.0×
LM head + final block	.672	92%	.937	2.8×
Final 2 blocks + LM head	**.678**	89%	.908	5.2×

Richer subspaces modestly improve matched utility correlation but cost substantially more. The LM-head implementation is therefore a cost–signal operating point, not the most accurate subspace.

A.12 END-TO-END COMPUTE ACCOUNTING

Experiments use up to 16 NVIDIA H100-SXM5-80GB GPUs: eight for FSDP training or gradient computation and eight for vLLM rollout service. The H100-hour figures below are *metered active-device time summed over accelerator processes*, not "allocated GPU count × wall-clock duration"; rollout and training services are not assumed to be simultaneously active at full utilization throughout the job. Code verification uses 128 CPU sandbox workers and is reported separately.

Table 12: **End-to-end selection and downstream compute.**

Component	Math	Code
40,000 candidate rollouts (H100-h)	6.42	9.16
Candidate gradients (H100-h)	2.31	2.22
$\mathcal{D}_v$, curvature, calibration (H100-h)	3.74	3.61
Ranking/I/O (H100-h)	.06	.07
End-to-end selection (H100-h)	**12.53**	**15.06**
Verifier CPU-hours	.28	91.4
Downstream GRPO (H100-h)	29.10	32.40
Full selection overhead	$\mathbf{.431}\times$	$\mathbf{.465}\times$
Selection + downstream	$1.431\times$	$1.465\times$

The arithmetic is $12.53/29.10 = .431$ for Math and $15.06/32.40 = .465$ for Code. Candidate selection therefore has nontrivial total cost despite using a small projected geometry.

Table 13: **Compute-matched random reference at $N = 100$.**

Domain	Method	Total budget	Performance
Qwen-Math	Random, 200 steps	1.00×	.675
Qwen-Math	Random, 286 steps	1.43×	.704
Qwen-Math	Random, 340 steps	1.70×	.716
Qwen-Math	Random, 400 steps	2.00×	.729
Qwen-Math	DataLens + 200 steps	1.431×	**.840**
Qwen-Code	Random, 200 steps	1.00×	.405
Qwen-Code	Random, 293 steps	1.465×	.428
Qwen-Code	Random, 340 steps	1.70×	.438
Qwen-Code	Random, 400 steps	2.00×	.447
Qwen-Code	DataLens + 200 steps	1.465×	**.458**

B Theoretical Analysis and Controlled-Probe Calibration

B.1 Marginal Candidate Scores and Subset Interactions

DataLens assigns a finite-step value to each candidate independently, whereas downstream GRPO trains a subset jointly. If, for analysis, a subset displacement were locally additive,

$$s_S = \sum_{i \in S} \alpha_i s_i,$$

then the second-order expansion gives

$$J_v(\theta_0 + s_S) - J_v(\theta_0) \approx \sum_{i \in S} \alpha_i g_v^\top s_i + \frac{1}{2} \sum_{i \in S} \alpha_i^2 s_i^\top H_v s_i + \sum_{i<j} \alpha_i \alpha_j s_i^\top H_v s_j. \tag{24}$$

The first two terms are marginal candidate contributions; the final term captures pairwise interactions and is absent from independent Top-N ranking. Actual GRPO is even less additive because of resampling, clipping, optimizer state, and parameter drift. DataLens is therefore a marginal screening method, and transfer from individual scores to jointly trained subsets is tested empirically.

B.2 Controlled-Probe Approximation Bound

Recall

$$U_{\mathrm{probe}}(s) = J_v(\theta + s) - J_v(\theta), \qquad \widehat{U}_{\mathrm{DL}}(s) = g_v^\top s + \frac{1}{2} s^\top H_v s.$$

Proposition 1. Assume that J_v is three-times continuously differentiable on the segment $\{\theta + ts : t \in [0,1]\}$ and that

$$\left| D^3 J_v(\theta + ts)[s,s,s] \right| \le M \|s\|^3$$

for all $t \in [0,1]$. Then

$$|U_{\mathrm{probe}}(s) - \widehat{U}_{\mathrm{DL}}(s)| \le \frac{M}{6} \|s\|^3.$$

Proof. Taylor's theorem gives some $\xi \in (0,1)$ such that

$$J_v(\theta + s) = J_v(\theta) + g_v^\top s + \frac{1}{2} s^\top H_v s + \frac{1}{6} D^3 J_v(\theta + \xi s)[s,s,s].$$

Subtracting the quadratic approximation and applying the assumption proves the claim.

B.3 Pairwise Ranking Preservation

Proposition 2. Under Proposition 1, the predicted and exact order of candidates i, j agree whenever the inequality in Eq. (13) holds.

Proof. Write $U_{\mathrm{probe}}(i) = \widehat{U}_{\mathrm{DL}}(i) + e_i$ with $|e_i| \le (M/6)\|s_i\|^3$. Then

$$|e_i - e_j| \le \frac{M}{6} (\|s_i\|^3 + \|s_j\|^3).$$

If the predicted gap exceeds this bound, the error cannot change its sign.

B.4 Why the Theory Does Not Imply Subset Optimality

Propositions 1 and 2 concern one controlled displacement at a time. Even under an additive local displacement model, Eq. (24) contains $s_i^\top H_v s_j$ for $i \neq j$. Actual GRPO further introduces resampling, optimizer state, clipping, and parameter drift. Thus the theory supports local marginal screening and pairwise probe ranking only; the bridge and downstream experiments provide separate evidence about transfer.

B.5 Matched Taylor Calibration

For $s_i = \eta d_i$, define

$$U_1(i) = g_v^\top s_i, \qquad Q_i = \frac{1}{2} s_i^\top H_v s_i, \qquad R_i = U_{\text{probe}}(i) - U_1(i).$$

The expansion predicts $R_i = Q_i + O(\|s_i\|^3)$, so the fitted slope between Q and R approaches one in the local regime.

Table 14: **Qwen-Math fixed-objective Q–R calibration.**

Update scale η	Calibration slope
1×10^{-7}	1.0002
1×10^{-6}	1.0012
1×10^{-5}	1.0113
5×10^{-5}	1.03
1×10^{-4}	0.97
2×10^{-4}	0.867
5×10^{-4}	0.612
1×10^{-3}	0.264
3×10^{-3}	0.038

Table 15: **Cross-setting Q–R calibration slopes.**

Setting	$\eta = 10^{-5}$	$\eta = 2 \times 10^{-4}$	$\eta = 10^{-3}$
Qwen-Math	1.01	.87	.26
DeepSeek-Math	.97	.84	.30
Mistral-Math	1.02	.82	.29
Qwen-Code	1.00	.86	.33
DeepSeek-Code	.98	.80	.35
Mistral-Code	1.03	.78	.38

The linear approximation error has fitted exponent $p_1 \approx 1.997$. After the quadratic term, the fitted exponent is $p_2 \approx 2.858$ and approaches three after excluding the numerical floor, consistent with the expected local orders.

C Additional Utility-Prediction Results

C.1 Measured Ranking Reversal

A representative Qwen-Math pair illustrates how the directional term can reverse a first-order ordering while matching the measured finite-step outcome.

Table 16: **Controlled-probe ranking reversal.** First-order alignment prefers Candidate A, whereas the directional correction reverses the DataLens ordering to match measured finite-step utility.

Quantity	Candidate A	Candidate B
U_1	2.40×10^{-2}	2.05×10^{-2}
$\frac{1}{2}\eta^2 q$	-1.82×10^{-2}	-0.31×10^{-2}
DataLens	0.58×10^{-2}	1.74×10^{-2}
Controlled U_{probe}	0.55×10^{-2}	1.69×10^{-2}
Real one-step bridge gain	0.49×10^{-2}	1.58×10^{-2}

Candidate A has the larger first-order score, but its curvature correction is substantially more negative. DataLens therefore ranks Candidate B above Candidate A, matching both measured U_{probe} and the realized one-step bridge gain.

C.2 Qwen-Math Audit-Tail Diagnostics

Table 17: **Qwen-Math audit-tail diagnostics.** Tail recovery and selected-set measured utility within the 441-candidate audit. Mean utility is standardized within the audit.

	Audit Recall@N		Mean measured utility (z)	
N	q	DataLens	q	DataLens
25	.520	**.640**	1.08	**1.33**
50	.600	**.720**	.91	**1.12**
100	.720	**.830**	.67	**.82**
200	.840	**.900**	.34	**.44**

At audit $N = 100$, DataLens recovers $83/100$ members of the measured-utility Top-100, compared with $72/100$ for q. Their predicted Top-100 sets share 68 candidates. The DataLens-only subset has mean measured probe utility .95, compared with .48 for the q-only subset; the corresponding full Top-100 selected-set means are .821 and .670. Because audit Top-100 is the top 22.7% of 441 measured candidates whereas full-pool Top-100 is roughly the top 3.1% of 3,262 eligible candidates, this is a measured-audit tail diagnostic rather than a full-pool oracle evaluation.

C.3 Paired Bootstrap Uncertainty

Table 18: **Paired-bootstrap differences in candidate-level Spearman correlation.** Positive values favor DataLens.

Setting	$\Delta(\text{DL} - U_1)$	95% CI	$\Delta(\text{DL} - q)$	95% CI
Qwen-Math	+.595	[.544, .645]	+.021	[−.009, .051]
DeepSeek-Math	+.503	[.447, .557]	+.039	[.011, .067]
Mistral-Math	+.465	[.409, .520]	+.033	[.006, .060]
Qwen-Code	+.460	[.408, .512]	+.032	[.008, .056]
DeepSeek-Code	+.445	[.389, .500]	+.035	[.009, .061]
Mistral-Code	+.425	[.368, .481]	+.035	[.008, .062]

C.4 AUDIT-SAMPLE TOP-100 DIAGNOSTICS

Table 19: **Audit-sample Top-100 overlap with exactly measured projected-probe utility.**

	Mathematical Reasoning			Code Generation		
Metric	Qwen	DeepSeek	Mistral	Qwen	DeepSeek	Mistral
q Audit Recall@100	.72	.68	.64	.66	.63	.60
DataLens Audit Recall@100	**.83**	**.76**	**.72**	**.75**	**.72**	**.69**

These values compare the Top-100 sets *within each measured audit*; because audit sizes differ and the full-pool Top-100 is much more extreme, they are not full-pool oracle recall estimates.

C.5 SENSITIVITY TO THE CONTROLLED UPDATE SCALE

Table 20: **Candidate-level Spearman correlation across update scales.**

Setting	10^{-4}	2×10^{-4}	4×10^{-4}	6×10^{-4}	8×10^{-4}
Qwen-Math	.612	.646	**.660**	.655	.638
DeepSeek-Math	.556	.579	**.587**	.584	.571
Mistral-Math	.505	.528	**.536**	.533	.522
Qwen-Code	.529	.551	**.558**	.555	.543
DeepSeek-Code	.500	.520	**.528**	.525	.514
Mistral-Code	.470	.490	**.497**	.494	.483
Mean	.529	.552	**.561**	.558	.545

Using 4×10^{-4} as the reference, average Top-100 overlap across the six settings is 80%, 92%, 100%, 92%, and 84% at the five scales in ascending order. On Qwen-Math, the corresponding overlaps are 77%, 91%, 100%, 93%, and 85%.

C.6 VALIDATION-SET SIZE ROBUSTNESS

Table 21: **Qwen-Math robustness to $|\mathcal{D}_v|$.**

$\lvert\mathcal{D}_v\rvert$	q Spearman	DataLens Spearman	DL Top-100 overlap	Downstream
64	$.594 \pm .023$	$\mathbf{.611 \pm .020}$	$.78 \pm .04$	$.819 \pm .014$
128	$.621 \pm .015$	$\mathbf{.641 \pm .013}$	$.89 \pm .03$	$.833 \pm .011$
256	.639	**.660**	1.00	$.840 \pm .010$

D Additional Downstream RL Results

D.1 External One-Shot Selector Comparison at $N = 100$

Table 22: **External one-shot selector comparison at $N = 100$.** Entries are three-seed means. LearnAlign/GradAlign rows are static scoring instantiations at θ_0, aligned with the one-shot selection setting used here.

	Mathematical Reasoning			Code Generation		
Method	Qwen	DeepSeek	Mistral	Qwen	DeepSeek	Mistral
Random	.675	.663	.635	.405	.383	.366
Difficulty $p(1-p)$	.699	.681	.651	.421	.398	.380
Gradient norm	.724	.696	.667	.435	.411	.393
LearnAlign (one-shot score)	.752	.711	.684	.448	.423	.405
GradAlign (one-shot score)	.756	.713	.686	.449	.425	.407
Empirical-Fisher	.765	.715	.687	.451	.428	.409
U_1	.701	.686	.657	.428	.404	.386
q	.714	.701	.671	.440	.415	.397
q_{dir}	.721	.707	.677	.446	.421	.403
DataLens	**.840**	**.720**	**.690**	**.458**	**.433**	**.415**

DataLens has the highest $N = 100$ point estimate in all six settings. The Qwen-Math gap is large, while the margins over empirical Fisher in the other five settings are only .003–.007 and are therefore treated as point estimates.

D.2 Probe–Downstream Reversal of Curvature Scores

In the measured-U_{probe} audits, raw directional curvature q has higher correlation than normalized curvature q_{dir} in all six settings, whereas q_{dir} has higher downstream mean performance than q in all 24 cells of Table 3. Because the controlled displacement is $s_i = \eta d_i$, the initial direction magnitude is part of the matched probe target and is retained by $q = d_i^\top H_v d_i$. By contrast, q_{dir} removes this magnitude and measures curvature per unit direction.

Over 200-step AdamW/GRPO training, minibatch averaging, coordinate-wise preconditioning, gradient clipping, repeated sampling, and parameter drift can weaken the correspondence between initial $\|d_i\|$ and cumulative training effect. Direction normalization can therefore reduce fidelity to the matched local probe while improving long-horizon subset behavior. Candidate-level finite-step utility and final downstream performance are related but distinct prediction targets.

D.3 Initial Policies and Full-Data Reference

Table 23: **Initial, Top-100, and full-data reference performance.**

Setting	Initial θ_0	Random $N = 100$	DataLens $N = 100$	Full 5,000
Qwen-Math	.658	.675	**.840**	.812
DeepSeek-Math	.620	.663	.720	**.748**
Mistral-Math	.586	.635	.690	**.718**
Qwen-Code	.374	.405	.458	**.482**
DeepSeek-Code	.348	.383	.433	**.456**
Mistral-Code	.324	.366	.415	**.438**

D.4 Primary and Additional Qwen-Math Seeds

Table 24: **Qwen-Math $N = 100$ across the three primary seeds.**

Method	Seed 42	Seed 7	Seed 1337	Mean ± std
Random	.668	.675	.682	$.675 \pm .007$
U_1	.694	.701	.708	$.701 \pm .007$
q	.706	.714	.722	$.714 \pm .008$
q_{dir}	.713	.721	.729	$.721 \pm .008$
DataLens	.830	.841	.849	$.840 \pm .010$

Five additional fixed-subset seeds produce DataLens values .836, .845, .832, .848, and .839. Pooling all eight training seeds gives DataLens $= .840 \pm .007$ and $U_1 = .702 \pm .005$. This experiment isolates training-seed variation only.

D.5 Selection-Seed and Full-Pipeline Robustness

Table 25: **Qwen-Math Top-100 DataLens robustness.** This table measures absolute DataLens stability rather than paired superiority under full-pipeline randomness.

Randomness varied	Runs	Accuracy	Selection overlap
Training seed only, fixed subset	8	$.840 \pm .007$	1.00
Rollout/scoring seed, fixed pool	9	$.836 \pm .010$	Jaccard $.78 \pm .04$
Pool × rollout × training	27	$.834 \pm .012$	not comparable across pools
Full-pipeline range	27	.811–.852	—

D.6 Explaining the Qwen-Math $N = 100$ Peak

Table 26: **Qwen-Math controls for the Top-100 peak.**

Configuration	Steps	Prompt draws	Draws/item	Accuracy
Top-100 main	200	1,600	16	$.840 \pm .010$
Top-200 main	200	1,600	8	$.751 \pm .012$
Top-200 exposure-matched	400	3,200	16	$.816 \pm .011$
Duplicated Top-100 multiset	200	1,600	16	$.838 \pm .009$
Ranks 101–200 only	200	1,600	16	$.704 \pm .012$
Top-100 + random tail	200	1,600	8	$.783 \pm .011$
Top-200, score-proportional	200	1,600	weighted	$.802 \pm .010$

D.7 Short-Horizon GRPO Protocol

For each model-domain setting, we draw 512 candidates by fixed hash ordering from the common nonzero-direction stratum and partition them into 64 mutually disjoint groups G of eight prompts. For candidate-level score $S_m(i)$, we use the group score defined in Eq. (14). This aggregation matches the eight-prompt granularity of a real GRPO minibatch. It averages marginal candidate scores rather than recomputing a geometric score for the aggregate displacement, so the cross-candidate terms in Eq. (24) are not explicitly included.

Every group starts from the same θ_0 and a fresh AdamW state. The same eight prompts form the minibatch at each of 20 GRPO steps, while eight new completions per prompt are sampled at every step. Scores use $\mathcal{D}_v$, whereas realized gain is measured on the independent $\mathcal{D}_{\mathrm{bridge}}$. Let θ_r denote the parameters after r optimizer steps; the bridge gain is defined in Eq. (15). This design evaluates transfer from the selection-time signal to real GRPO optimization rather than remeasuring the score-defining objective on the same data.

D.8 SHORT-HORIZON GRPO CONSISTENCY

Table 27: **Setting-level bridge to real GRPO.** Spearman correlation with realized gain on the independent bridge split.

		Math			Code			
r	Method	Qwen	DeepSeek	Mistral	Qwen	DeepSeek	Mistral	Mean
1	U_1	.12	.18	.15	.20	.17	.14	.16
1	q	.52	.43	.41	.42	.39	.38	.43
1	q_{dir}	.43	.41	.39	.42	.38	.37	.40
1	DataLens	**.56**	**.55**	**.53**	**.54**	**.51**	**.50**	**.53**
5	U_1	.10	.16	.13	.18	.15	.12	.14
5	q	.48	.40	.38	.39	.37	.35	.40
5	q_{dir}	.43	.38	.35	.37	.35	.34	.37
5	DataLens	**.53**	**.52**	**.50**	**.51**	**.48**	**.47**	**.50**
10	U_1	.07	.13	.10	.15	.12	.09	.11
10	q	.40	.36	.34	.35	.32	.31	.35
10	q_{dir}	.39	.34	.31	.33	.31	.30	.33
10	DataLens	**.47**	**.47**	**.45**	**.46**	**.43**	**.43**	**.45**
20	U_1	.04	.10	.07	.11	.09	.07	.08
20	q	.32	.29	.27	.28	.26	.24	.28
20	q_{dir}	.32	.27	.25	.26	.24	.22	.26
20	DataLens	**.37**	**.40**	**.39**	**.39**	**.37**	**.35**	**.38**

For Qwen-Math, paired DataLens–q differences are $+.04$ at one step (95% CI $[.005, .078]$), $+.05$ at five steps ($[.010, .092]$), $+.07$ at ten steps ($[.021, .119]$), and $+.05$ at twenty steps ($[-.008, .108]$). The 20-step interval includes zero.