WHICH QUESTIONS ARE WORTH THINKING ABOUT? CROSS-DEVELOPER TRANSFER OF THINKING VALUE

Anonymous authors
Paper under double-blind review

ABSTRACT

Which questions are worth the additional cost of thinking? We test whether question-level accuracy gains from configured thinking modes transfer across model developers. We define *thinking value* as the paired accuracy change between configured thinking and non-thinking modes and evaluate 18 models from seven developers on 300 MMLU-Pro STEM questions and 198 GPQA-Diamond questions under three prompting strategies. Gains estimated from other developers predict a held-out developer's gains after IRT-style adjustment for measured difficulty and discrimination, with pooled correlations of 0.264 and 0.273. Transfer persists with an independent difficulty panel and under request-setting and scoring checks. In retrospective allocation on the evaluated questions, ranking transferred gains per predicted additional token recovers 91.8% and 79.1% of the always-thinking accuracy gain at half of the always-thinking additional-token budget, compared with 83.5% and 74.4% for a difficulty-based gain predictor per token. The advantage is statistically resolved on MMLU-Pro but not GPQA-Diamond. An ability-shift simulation produces comparable transfer, leaving the source of this predictability unresolved. The tested text-based predictors generalize weakly to unseen questions. These findings characterize cross-developer transfer of thinking value on previously evaluated questions and motivate learning to predict this value for adaptive computation.

1 INTRODUCTION

Inference-time reasoning can improve language-model accuracy, but its token cost and benefit vary across questions. Models may generate long traces for questions they can already answer directly, sometimes reducing accuracy (Liu et al., 2025; Su et al., 2025; Zhou et al., 2026; Sui et al., 2025). This uneven return makes balancing accuracy and computation difficult (Aggarwal et al., 2025). Test-time scaling methods address the allocation problem (Snell et al., 2024; Wu et al., 2025; Huang et al., 2026), and recent systems learn when individual models should think (Fang et al., 2025; Tu et al., 2025; Zhang et al., 2025; Kassenaar et al., 2026). Can evaluations from other developers predict which questions benefit from enabling thinking in a held-out developer's models?

A difficult question does not necessarily benefit from thinking: switching modes may still leave the answer incorrect. Conversely, thinking can introduce errors on questions answered correctly without it. We define *thinking value* as the signed accuracy change between configured thinking and non-thinking modes, averaged over a specified distribution of models and prompts. Difficulty describes correctness under one policy; thinking value describes the change between policies. If these gains transfer across developers, existing evaluations can help identify where another developer should spend additional computation.

We test transfer by holding out entire developers rather than individual model variants. Eighteen reasoning-capable open models from seven developers answer the same 300 MMLU-Pro STEM questions and all 198 GPQA-Diamond questions in both modes, under zero-shot, few-shot, and explicit chain-of-thought (CoT) prompts. Gains are averaged within developers, with equal weight assigned to each developer. In leave-one-developer-out (LODO) evaluation, gains from six developers predict those of the seventh. All 35 complementary three-versus-four developer partitions provide a second test of agreement (Section 4). These tests reuse evaluations of the same questions; a separate experiment holds out questions as well.

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.

Our contributions are:

1. **Predicting thinking gains for held-out developers.** Gains measured for other developers predict a held-out developer's gains on MMLU-Pro and GPQA-Diamond. This transfer persists after the fitted difficulty and discrimination adjustments and under independent-difficulty, prompt, request-setting, and scoring checks.
2. **Transferred gains inform token-budget allocation.** On previously evaluated questions, transferred gains improve retrospective allocation relative to gains predicted from difficulty. With both scores normalized by predicted token cost, value ranking recovers 91.8% and 79.1% of the always-thinking gain at half its additional-token budget, compared with 83.5% and 74.4% for the difficulty-fit baseline. The advantage is statistically resolved on MMLU-Pro but not GPQA-Diamond.
3. **Testing explanations and generalization.** An ability-shift simulation produces comparable adjusted transfer without a thinking-specific item parameter (Appendix A.18), while the tested text-based predictors generalize inconsistently to unseen questions. These analyses distinguish the demonstrated cross-developer transfer from its explanation and its extension to new questions.

2 Related Work

Elicited and native reasoning. CoT prompting elicits intermediate steps and improves many reasoning tasks (Wei et al., 2022; Kojima et al., 2022), and self-consistency extends it through repeated sampling (Wang et al., 2023). Reasoning models also expose native controls over reasoning effort (Guo et al., 2025; OpenAI et al., 2026). These controls can change generation policy, token budgets, and provider settings. We study the resulting configured-mode contrast and document its request settings.

When reasoning helps, and when it hurts. Sprague et al. (2025) show that CoT helps mainly on mathematics and symbolic reasoning, and probability, memorization, and noisy intermediate computation modulate its efficacy (Prabhakar et al., 2024). Reasoning can also hurt: CoT lowers accuracy on task families where deliberation impairs humans (Liu et al., 2025), controlled-complexity studies find a regime where non-reasoning models beat reasoning models (Shojaee et al., 2025), and overthinking, additional reasoning that increases cost without improving accuracy or changes a correct answer to an incorrect one, has been documented (Su et al., 2025; Zhou et al., 2026; Sui et al., 2025; Zhu & Li, 2025; Aggarwal et al., 2025). Our work shifts the unit of analysis from task averages and complexity regimes to the paired return on a specific item, and tests whether that item ranking transfers to developers excluded from its estimation.

Adaptive test-time compute. Adaptive test-time computation allocates resources through repeated sampling, verification, and search (Cobbe et al., 2021; Lightman et al., 2024; Brown et al., 2024), or through difficulty and confidence estimates (Snell et al., 2024; Huang et al., 2026). Other methods learn when to use reasoning (Fang et al., 2025; Tu et al., 2025; Zhang et al., 2025), select among reasoning modes (Kassenaar et al., 2026), or allocate budgets using latent predictions, item response models, and constrained policies (Li et al., 2026b; Fernandez et al., 2026; Zhai et al., 2026). Reasoning length can also be controlled through budget forcing, reinforcement learning, and trace pruning (Muennighoff et al., 2025; Aggarwal & Welleck, 2025; Luo et al., 2026). A related literature routes queries between models (Ong et al., 2025; Chen et al., 2024). Despite these advances, models asked to ration a shared budget across questions can allocate it largely by presentation order (Fan et al., 2026). We study the question-level gains that motivate allocation and ask whether they transfer across developers.

Cross-model regularities. Benchmark performance exhibits regularities across model families (Ruan et al., 2024), and models from different developers can make correlated errors (Goel et al., 2025; Kim et al., 2025). These findings motivate testing whether agreement in thinking gains remains after accounting for measured correctness patterns. Closest to our question, Li et al. (2026a) validate their reasoning-intensive filtering by showing that four frontier models, including one from another family, largely agree on which instances flip from incorrect at low reasoning effort to correct

at high effort; Cheng et al. (2026) study a different sense of transfer, reusing one model's reasoning trace inside another. We examine signed gains, counting both corrections and regressions, and evaluate prediction with entire developers held out. We then ask whether those gains retain predictive information after fitted correctness adjustments and improve allocation under a token budget. In our comparison with a donor positive-flip predictor, the allocation point estimates favor signed gains (Appendix B).

IRT and efficient evaluation. IRT models benchmark responses through respondent ability and item parameters such as difficulty and discrimination (Lord & Novick, 1968); these regularities support estimating scores from small item subsets (Polo et al., 2024), compressing benchmarks (Kipnis et al., 2025), and routing questions among models (Song et al., 2025). These methods model correctness under an inference policy. We study the difference in accuracy between two policies and test whether it remains predictable across developers after adjustment for difficulty and discrimination estimated from non-thinking responses.

Benchmarks and transfer. MMLU-Pro increases the answer set and difficulty of MMLU (Hendrycks et al., 2021) while reducing prompt sensitivity (Wang et al., 2024); GPQA-Diamond tests graduate-level science (Rein et al., 2023). The primary analysis uses six equally sized MMLU-Pro STEM categories; GPQA-Diamond supplies an item-level replication with a fixed four-option format and a different category distribution.

3 Experimental Design

Models and native modes. Figure 1 summarizes the design. The evaluation covers 28 open instruction-tuned models. Eighteen expose paired native thinking and non-thinking modes and form the gain analysis: DeepSeek-V3.2 (DeepSeek-AI et al., 2025), GLM-5 (GLM-5-Team et al., 2026), Kimi-K2.5 (Team et al., 2026b), Mistral-Small-4, two Gemma-4 models (Team et al., 2026a), two GPT-OSS models (OpenAI et al., 2025), and five Qwen3 plus five Qwen3.5 models (Yang et al., 2025). These map to seven developers: Alibaba, DeepSeek, Google, Mistral, Moonshot, OpenAI, and Zhipu. GPT-OSS low effort counts as non-thinking and high effort as thinking (medium is excluded); low effort still reasons briefly, so for these two models the label is operational (Appendix A). Ten non-thinking-only models are reserved for the outcome-independent difficulty panel and contribute no thinking-gain estimates.

Questions, prompts, and scoring. The primary set is a seed-42 stratified random sample of 300 MMLU-Pro questions, 50 each from biology, chemistry, computer science, engineering, mathematics, and physics, drawn before any model evaluation; 265 have ten choices and 35 have three to nine. The replication set contains all 198 four-choice GPQA-Diamond questions. For each benchmark, the paired design covers 18 models, every question (300 or 198), three prompt strategies, and two modes. The prompts are zero-shot, few-shot, and explicit CoT. We analyze archived responses generated through hosted APIs with logged request settings. A fixed auxiliary model (`gpt-5.4-mini`) ex-

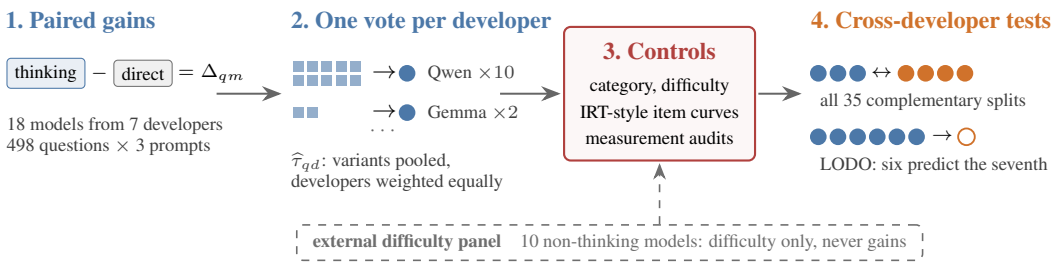

Figure 1: Measurement and validation design. Paired gains (1) are averaged within developers, giving each developer equal weight (2). Controls (3) and transfer tests (4) use the developer partitions described in Section 3. The ten-model external panel supplies difficulty estimates and does not contribute paired gains.

tracts final answers, which are scored by normalized exact match. Missing or unparseable answers are counted as incorrect.

Paired gain and thinking value. Let Y_{qmsz} be correctness for question q, model m, prompt strategy s, and mode $z \in \{0, 1\}$, after averaging repeated calls where present. The empirical paired gain between the two configured modes is

$$\Delta_{qms} = Y_{qms1} - Y_{qms0}. \tag{1}$$

We average over the three shared prompts to obtain Δ_{qm}, and over each developer's model variants so every developer receives equal weight:

$$\widehat{\tau}_{qd} = \frac{1}{|\mathcal{M}_d|} \sum_{m \in \mathcal{M}_d} \Delta_{qm}. \tag{2}$$

Thinking value τ_q is the expected paired gain under a specified distribution of developers and prompts. We estimate it by averaging the three prompts within each model, models within each developer, and the seven developers with equal weight: $\widehat{\tau}_q = \frac{1}{7} \sum_d \widehat{\tau}_{qd}$. The contrast includes any provider-setting changes associated with the configured mode. Inference is conditional on the selected developer roster and prompt protocol.

Developer-separated tests. Holding out an individual model can leave other variants from the same developer among the donors; we therefore hold out entire developers to test whether the gain signal extends beyond within-developer resemblance. Two complementary designs are used. First, all $\binom{7}{3} = 35$ splits place three developers in group A and the remaining four in group B, and we correlate their equally weighted question gains. Second, seven LODO folds use the mean of six developers as the item predictor for the seventh, compared against a category baseline and a stronger baseline that fits a category-specific quartic function of non-thinking solve rate on training developers only.

Adjustment and uncertainty. Shared difficulty could generate agreement in gains. We therefore test whether transfer persists after adjustment for measured correctness patterns. For developer splits, we residualize gains separately within each group against category indicators and category-specific quartic functions of non-thinking solve rate. For LODO, we subtract the same donor-fitted baseline from the item predictor and held-out gain, then pool the residual pairs across folds. Confidence intervals for the transfer statistics use a 2,000-replicate question-cluster bootstrap (Efron, 1979), retaining all seven held-out observations for each sampled question; resampling whole questions makes the intervals robust to within-question correlation across developers, prompts, and modes (Colin Cameron & Miller, 2015). Intervals are conditional on the fitted controls and evaluated roster, and the 35 partitions overlap. We use the question-cluster bootstrap rather than the stratified permutation procedure, whose rejection rates were inflated under the overlapping-donor design in null simulations (Appendix A.18).

IRT-style correctness control and reference correlations. The IRT-style sensitivity augments the category-specific quartic difficulty control with estimated item discrimination and its square. Discrimination is obtained from regularized two-parameter logistic fits to non-thinking responses, treating model–prompt pairs as respondents and conditioning on ability estimated from overall non-thinking accuracy (Appendix A). We also use a two-way random-effects model to estimate a reference correlation between a six-developer mean and one developer under exchangeability assumptions (Cronbach et al., 1972). Repeated AIME runs provide a separate estimate of sampling variability on that benchmark (Appendix A.12).

Independent difficulty and protocol sensitivities. Because the non-thinking response contributes to both the gain estimate and the usual difficulty control, we also measure difficulty with a separate panel that contributes no paired gains: we average non-thinking correctness over ten additional models and estimate its split-half reliability. This external difficulty replaces paired-developer solve rate in every split and LODO baseline, with a stricter version using only six Meta and Microsoft models absent from the paired roster. Additional checks exclude questions with any flagged extraction failure or truncation, adjust for MMLU-Pro answer-option counts, and restrict the analysis to seven models whose logged sampling fields match across modes.

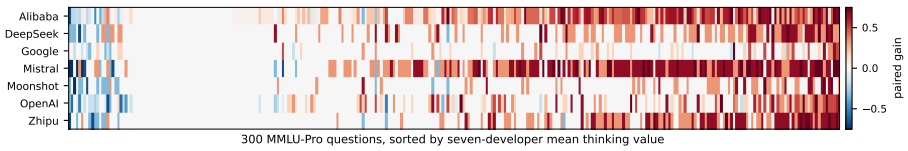

Figure 2: Developer-level paired gains on MMLU-Pro. Rows represent developers; columns represent questions ordered by their seven-developer mean gain. Similar colors within a column indicate agreement in gain direction. This visualization is descriptive; Figure 3 tests transfer with developers held out. Figure 6 in Appendix B shows the corresponding GPQA-Diamond results.

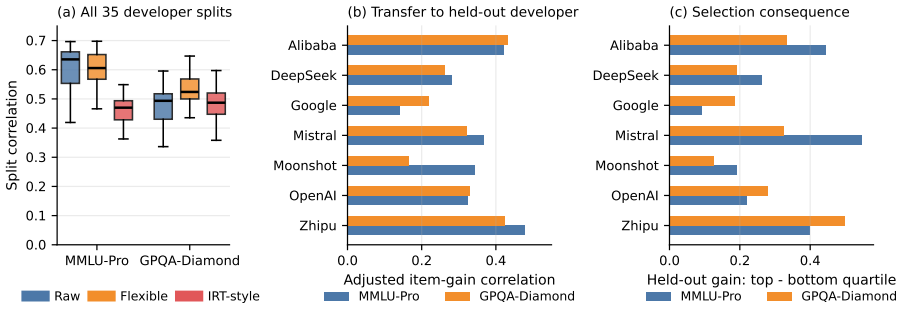

Figure 3: Thinking value transfers on both benchmarks. (a) Raw and adjusted correlations for all 35 developer splits. (b) Flexible-adjusted transfer to each held-out developer. (c) Held-out gain difference between the predicted top and bottom quartiles.

4 Thinking Value Transfers Across Developers

4.1 Disjoint developer groups agree

Figure 2 displays paired gains for each developer and question. Across the 35 three-versus-four splits on MMLU-Pro, the median raw Pearson correlation between group-level question gains is 0.636 (Spearman 0.669). Removing category means changes it only to 0.609; removing the flexible category-specific difficulty functions leaves 0.606 (range 0.466–0.698), so the agreement goes beyond subject areas and the fitted difficulty trends.

Questions in group A's predicted top quartile have a mean gain in group B that is 0.137 above group B's all-question mean; reversing the groups gives 0.142. Selection uses gains from the other group only. Three items selected by fixed rules illustrate the contrast. A chemistry question has a seven-developer mean gain of $+0.633$; an engineering question has a gain of -0.476, with no developer improving; and another engineering question has a non-thinking solve rate of 0.067 but a gain of only $+0.014$. Appendix B gives the question texts and selection rules.

4.2 Transfer to an unseen developer

Gains from six developers predict the seventh developer's gains on MMLU-Pro, with a pooled correlation of 0.306 after the donor-fitted category-and-difficulty adjustment (95% question-cluster bootstrap CI $[0.230, 0.371]$). The item predictor reduces summed squared error by 8.0% relative to the flexible baseline ($[3\%, 13\%]$) and by 12.4% relative to category alone ($[7\%, 17\%]$), improving prediction for six of seven developers. Raw correlations are positive for every developer; Table 1 reports the individual estimates. Unlike the split analysis, which compares group averages, LODO predicts one developer's gains.

4.3 Cross-developer transfer replicates on GPQA-Diamond

The same protocol yields positive transfer across the 198 GPQA-Diamond questions (Table 2). The pooled LODO correlation after flexible adjustment is 0.285 (95% question-cluster bootstrap CI $[0.177, 0.374]$). Raw LODO correlations are positive for all seven developers (range 0.193–0.510;

Table 1: Transfer to a held-out developer on MMLU-Pro. r: raw item correlation; r_{adj}: after the training-fitted quartic difficulty trend; Δ_Q: held-out gain, predicted top minus bottom quartile.

Held-out developer	Models	r	r_{adj}	Δ_Q
Alibaba	10	0.619	0.421	0.445
DeepSeek	1	0.411	0.281	0.264
Google	2	0.283	0.141	0.092
Mistral	1	0.504	0.367	0.549
Moonshot	1	0.363	0.345	0.191
OpenAI	2	0.396	0.326	0.220
Zhipu	1	0.570	0.479	0.400
Median	–	0.411	0.345	0.264

Table 2: Cross-benchmark replication and targeted controls. Split statistics are medians over the 35 complementary developer partitions; pooled LODO correlations use residuals from the indicated training-only control, with question-clustered CIs. Δ_Q: median held-out top-minus-bottom predicted-quartile gain.

Benchmark	Items	Split raw	Split flexible	Split IRT	Pooled LODO IRT [95% CI]	Δ_Q
MMLU-Pro	300	0.636	0.606	0.470	0.264 [0.201,0.319]	0.264
GPQA-Diamond	198	0.494	0.524	0.487	0.273 [0.170,0.361]	0.280

median 0.335), and the median held-out gain difference between predicted top and bottom quartiles is 0.280. Across the 35 developer splits, the median correlation is 0.494 before adjustment and 0.524 after adjustment for subdomain and non-thinking solve rate. Transfer thus extends to a second multiple-choice benchmark with a different option count, item pool, and category distribution.

4.4 Transfer after correctness adjustment

Models from different developers often err on the same items (Goel et al., 2025; Kim et al., 2025). We therefore augment the difficulty control with estimated item discrimination (Appendix A gives the explicit specification). On MMLU-Pro, this reduces the median split correlation from 0.606 to 0.470, while all splits remain positive (range 0.363–0.549). The pooled LODO correlation is 0.264 (95% CI $[0.201, 0.319]$), and item-level prediction reduces MSE by 5.5% relative to the IRT-style baseline ($[2\%, 9\%]$), improving six of seven developers. On GPQA-Diamond, the median split correlation is 0.487 and the pooled LODO correlation is 0.273 ($[0.170, 0.361]$). The corresponding MSE reduction is 4.3%, but its interval includes zero ($[-2\%, 11\%]$).

We also estimate difficulty using a separate ten-model panel that contributes no paired gains. Replacing the paired-model solve rate with this external estimate leaves all adjusted splits positive. Pooled LODO correlations are 0.316 ($[0.248, 0.374]$) on MMLU-Pro and 0.287 ($[0.194, 0.361]$) on GPQA-Diamond. A stricter panel containing only developers absent from the paired roster gives similar results (Appendix A.9).

Do the remaining correlations require a thinking-specific item parameter? In an ability-shift simulation, each model receives an overall ability change along fitted item-response curves, with no thinking-specific parameter assigned to questions. The median pooled IRT-adjusted correlations are 0.243 on MMLU-Pro and 0.323 on GPQA-Diamond, compared with observed values of 0.264 and 0.273 (Appendix A.18). Thus, shared response-curve structure can produce transfer of comparable magnitude. The fitted adjustments establish predictive information after the selected controls, but do not distinguish that explanation from a separate thinking-specific item dimension.

Variation across questions and developers. In the MMLU-Pro model–question matrix, question main effects account for 30.4% of observed gain variance, model main effects for 9.7%, and interaction plus sampling noise for the remainder (Figure 7). Appendices A.11, A.12, and A.15 examine reference correlations, variance components, and repeated-run reliability. The repeat-based noise estimates apply to AIME; sampling noise is not separately estimated on the two primary benchmarks.

5 ROBUSTNESS

Agreement across prompt strategies. Equal-developer question effects correlate across every pair of prompt strategies ($r = 0.535$–0.660, with similar within-category values; Table A11). Holding out both prompt strategy and developer group gives median correlations of 0.376 before adjustment and 0.406 after flexible adjustment. Transfer therefore extends across the tested prompt strategies.

Transfer with matched logged sampling fields. Eleven models change at least one logged sampling field between modes; seven models spanning five developers keep identical request-level temperature, top-p, and top-k (DeepSeek-V3.2, GLM-5, Mistral-Small-4, both GPT-OSS, both Gemma-4). All ten two-versus-three developer splits on this subset remain positive, with median adjusted $r = 0.526$ on MMLU-Pro and 0.504 on GPQA, and pooled adjusted LODO correlations of 0.255 ($[0.170, 0.330]$) and 0.268 ($[0.152, 0.361]$). The system-message difference between modes and any server-side defaults remain part of the treatment (Appendix H).

Transfer after excluding flagged responses. Removing every question with an extraction or truncation flag retains 267 MMLU-Pro and 165 GPQA-Diamond items. All 35 adjusted splits remain positive, with median correlations of 0.586 and 0.569; pooled adjusted LODO correlations are 0.273 (95% CI $[0.175, 0.361]$) and 0.309 ($[0.187, 0.411]$). Adjusting for MMLU-Pro answer-option counts gives a median split correlation of 0.600 and pooled LODO correlation of 0.298. Appendix A reports the filtering procedure and response-level flag rates.

Sensitivity to model composition. Transfer also persists when each developer contributes a single model variant. Across all 40 such rosters, median pooled LODO correlations are 0.26 on MMLU-Pro and 0.25 on GPQA-Diamond, with minima of 0.22 and 0.20. Excluding the GPT-OSS models, whose contrast compares low with high reasoning effort rather than disabling reasoning, gives pooled LODO correlations of 0.28 and 0.27 (Appendix A.9). These checks show that positive transfer persists under both changes to the evaluated roster.

6 ALLOCATION UNDER MATCHED TOKEN BUDGETS

Can cross-developer prediction improve allocation at a fixed token budget? We compare rankings based on donor-measured thinking value with rankings based on gains predicted from donor-measured difficulty. Both use the same evaluations from the other six developers. The difficulty-fit baseline uses a category-specific quartic function of non-thinking solve rate. We evaluate both scores directly and after dividing by donor-predicted additional completion tokens.

Each ranking is evaluated against the held-out developer's realized gains and additional completion-token costs. These costs determine retrospective budget usage but do not enter the donor-based ranking (Appendix A.13). Allocation confidence intervals use a separate 1,000-replicate question-cluster bootstrap that resamples questions and re-sorts the fixed donor scores without refitting the scores or difficulty surfaces (Appendix A.13).

At half the additional completion-token budget used by always thinking, value per predicted token recovers 91.8% of the always-thinking accuracy gain on MMLU-Pro, compared with 83.5% for difficulty-fit gain per token. On GPQA-Diamond, the corresponding values are 79.1% and 74.4% (Figure 4). The absolute accuracy advantages are 1.1 percentage points on MMLU-Pro (95% CI $[0.5, 1.9]$) and 1.0 percentage point on GPQA-Diamond ($[-0.2, 2.1]$); the latter interval includes zero.

Without cost normalization, value ranking recovers 80% and 68% of the gain, compared with 59% and 53% for difficulty-fit ranking. The absolute accuracy advantages are 2.9 percentage points ($[0.8, 4.0]$) and 3.5 percentage points ($[1.0, 5.7]$). These gain fractions average developer-specific ratios; dividing pooled absolute gains instead gives 75% and 66% for value ranking. Hardest-first ranking recovers only 36% and 39%, below simulated random allocation at the evaluated budgets up to three quarters of the additional-token cost. Table A6 reports the policy results at fixed budgets.

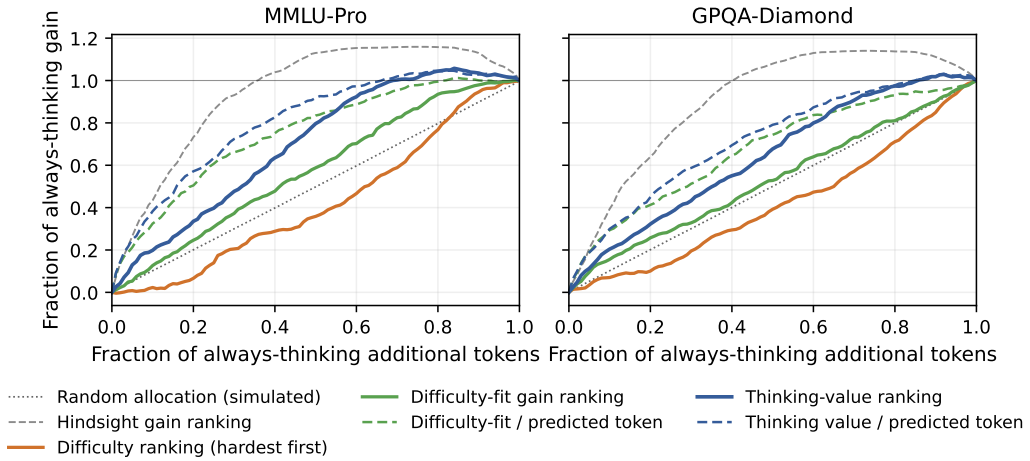

Figure 4: Retrospective allocation curves, averaged equally over held-out developers. Rankings use the other six developers; costs use the held-out developer's realized additional completion tokens. Difficulty-fit scores predict gains from measured difficulty, and dashed curves divide scores by predicted additional tokens. Random allocation uses the same evaluation rule. Hindsight ranks by realized gain and is not a cost-optimal allocation.

Accounting for regressions. We also compare signed value with the donor mean rate of incorrect-to-correct flips. Without cost normalization, signed value recovers 80% and 68% of the always-thinking gain at half the additional-token budget, compared with 70% and 60% for the positive-flip predictor. These allocation point estimates favor accounting for regressions as well as improvements (Appendix B).

A fixed positive-value rule provides a simple operating point: enable thinking only when its donor-predicted gain is positive. Relative to always thinking, this rule reduces additional-token use by 23% on MMLU-Pro and 14% on GPQA-Diamond, while recovering gain fractions of 1.03 (95% CI $[0.98, 1.12]$) and 1.01 ($[0.98, 1.06]$). Gain fractions above one are possible because a selective policy can omit questions on which thinking reduces accuracy. These estimates retain approximately the full gain, with intervals that permit small losses. Applying the same rule to difficulty-fit predictions uses 95% and 99% of the always-thinking additional tokens. Appendix E reports a separate category-based analysis.

Dependence on donor-panel size. We repeat the analysis using all subsets of one to six available donor developers (Appendix B). With one donor, the median adjusted held-out correlation is 0.19 on both benchmarks; with six donors, it is 0.34 on MMLU-Pro and 0.32 on GPQA-Diamond. In this sensitivity analysis, median gain recovery at half the additional-token budget increases from 65% to 77% on MMLU-Pro and from 60% to 68% on GPQA-Diamond as the panel grows from one donor to six. Transfer is therefore observable with a smaller panel, although acquiring the donor evaluations remains outside the reported allocation budget.

Can question text replace prior evaluations of the same item? We test this by holding out both a developer and a question fold. A ridge predictor using category, structural counts, and TF-IDF features achieves a median developer-level correlation of 0.129 and reduces MSE by 0.9% relative to category alone. A ridge predictor using cached, frozen 1,024-dimensional text embeddings yields pooled correlations of 0.131 on MMLU-Pro (95% CI $[0.067, 0.188]$) and -0.115 on GPQA-Diamond ($[-0.179, -0.045]$). The tested text predictors therefore do not provide consistent generalization to new questions. The successful allocation experiments above use donor evaluations of the target questions.

7 Discussion

Reusing evaluations across developers. Evaluations of one developer's models can inform thinking decisions for another developer's models. On the same question set, donor gains predict a held-

out developer's gains and improve the cost-aware allocation comparison on MMLU-Pro, with an unresolved advantage on GPQA-Diamond. Transfer between the evaluated Qwen3 and Qwen3.5 model sets also shows that gain rankings can carry across these two generations (Appendix B). Such reuse could be useful when a question set is evaluated repeatedly across models. The present allocation experiment measures the value of donor information once available; deployment would also need to account for the cost of acquiring it.

Why developer separation matters. Holding out developers prevents variants from the same developer from appearing on both sides of a transfer test. Performance still varies across developers and is weakest for Google on MMLU-Pro. AIME estimates are less reliable for Google than for Alibaba, but unequal variant counts and the different benchmark prevent attributing the primary-benchmark difference to sampling noise alone.

Dependence on the evaluation protocol. The AIME analysis shows that measured transfer is sensitive to the reasoning and truncation regime. Transfer is positive under external difficulty controls, but the estimate is near zero after truncated responses are removed (Appendix F). Because this filtering changes the evaluated responses and items, it does not identify the effect of token limits. Experiments with assigned budgets could distinguish budget effects from other features of the configured-mode contrast (Muennighoff et al., 2025; Aggarwal & Welleck, 2025). Truncation is rare on the two primary multiple-choice benchmarks.

Testing why gains transfer. One hypothesis is that thinking helps more with failed derivations than with missing task knowledge, while additional reasoning can also introduce regressions. The category differences and prompt-substitution results in Appendices C and D motivate testing this account, but do not distinguish it from alternative explanations. A follow-up experiment could label these error types and measure their gains under assigned reasoning budgets, testing whether the same error types benefit across developers.

Limitations. The primary evidence covers two multiple-choice benchmarks, three prompts, and seven selected developers, four represented by a single paired model. Confidence intervals are conditional on this roster and the fitted controls. Neither primary benchmark has repeated calls within a model–question–prompt–mode cell, so sampling reliability is not directly estimated. Answer extraction has not been independently validated by humans, and GPQA-Diamond is a retrospective replication.

The measured contrast includes system-message and possible provider-setting changes (Appendix H). Its transfer is compatible with an ability-shift response-curve explanation (Appendix A.18). The allocation experiments evaluate previously assessed questions using completion-token costs; generalization to new questions or developers, assigned reasoning budgets, and end-to-end deployment costs requires separate evaluation.

8 Conclusion

Thinking gains measured for one set of developers predict gains for another on the same questions. This transfer has an allocation consequence: at matched additional-token budgets, cost-aware value ranking improves on the difficulty-fit baseline on MMLU-Pro, with an unresolved advantage on GPQA-Diamond. Shared response curves remain a viable explanation, and the tested text predictors do not generalize consistently to new questions. The results motivate a specific objective for adaptive computation: predict the accuracy gained by switching modes, and test that prediction on unseen questions under assigned reasoning budgets.

Ethics Statement

The study uses public benchmarks and archived model outputs and involves no human subjects or private data. Adaptive reasoning may reduce computation, but allocation errors could disproportionately affect particular topics or linguistic styles. Any deployment should evaluate performance across these groups and under distribution shift.

Reproducibility Statement

The paper and Appendix A document the protocol in the detail needed for independent reproduction: data inclusion filters, question selection (a seed-42 stratified sample), the model roster with provider identifiers and logged request settings, developer mapping and aggregation, the control specifications including the explicit baseline in Equation 3, split and LODO construction, bootstrap and permutation procedures with replicate counts, and the allocation evaluation rules. Both benchmarks are public and the evaluated models are publicly accessible, so the measurement protocol can be rerun from this description. Our analysis is offline from archived responses and makes no new model calls (one appendix baseline reads previously cached text embeddings); randomized bootstrap and permutation routines use fixed seeds, and a verification script checks reported numerical values against the saved analysis outputs. The complete analysis code, saved outputs, and the task and answer-extraction prompt templates will be released upon publication.

AI Use Statement

The authors used generative AI tools for grammar checking, text editing, feedback, and code inspection. The authors recomputed the reported results from archived outputs and take responsibility for the analyses, citations, and manuscript. These research-assistance uses were separate from the model evaluations and the auxiliary answer-extraction model described in Section 3.

References

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, James R. Lee, Yin Tat Lee, Yuanzhi Li, Weishung Liu, Caio C. T. Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Xin Wang, Rachel Ward, Yue Wu, Dingli Yu, Cyril Zhang, and Yi Zhang. Phi-4 technical report, 2024. URL https://arxiv.org/abs/2412.08905.

Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=4jdIxXBNve.

Pranjal Aggarwal, Seungone Kim, Jack Lanchantin, Sean Welleck, Jason Weston, Ilia Kulikov, and Swarnadeep Saha. Optimalthinkingbench: Evaluating over and underthinking in llms, 2025. URL https://arxiv.org/abs/2508.13141.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling, 2024. URL https://arxiv.org/abs/2407.21787.

William Brown. Some experimental results in the correlation of mental abilities. *British Journal of Psychology*, 3:296–322, 1910.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=cSimKw5p6R. Featured Certification.

Xinyuan Cheng, Beiduo Chen, Philipp Mondorf, and Barbara Plank. Reasoning that travels: Dissecting how chain-of-thought transfers across models, 2026. URL https://arxiv.org/abs/2605.28913.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

A. Colin Cameron and Douglas L. Miller. A practitioner's guide to cluster-robust inference. *Journal of Human Resources*, 50(2):317–372, 2015. ISSN 0022-166X. doi: 10.3368/jhr.50.2.317. URL https://jhr.uwpress.org/content/50/2/317.

Lee J. Cronbach, Goldine C. Gleser, Harinder Nanda, and Nageswari Rajaratnam. *The Dependability of Behavioral Measurements: Theory of Generalizability for Scores and Profiles*. Wiley, 1972.

DeepSeek-AI, Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenhao Xu, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Erhang Li, Fangqi Zhou, Fangyun Lin, Fucong Dai, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Hao Li, Haofen Liang, Haoran Wei, Haowei Zhang, Haowen Luo, Haozhe Ji, Honghui Ding, Hongxuan Tang, Huanqi Cao, Huazuo Gao, Hui Qu, Hui Zeng, Jialiang Huang, Jiashi Li, Jiaxin Xu, Jiewen Hu, Jingchang Chen, Jingting Xiang, Jingyang Yuan, Jingyuan Cheng, Jinhua Zhu, Jun Ran, Junguang Jiang, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kaige Gao, Kang Guan, Kexin Huang, Kexing Zhou, Kezhao Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Wang, Liang Zhao, Liangsheng Yin, Lihua Guo, Lingxiao Luo, Linwang Ma, Litong Wang, Liyue Zhang, M. S. Di, M. Y Xu, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Panpan Huang, Peixin Cong, Peiyi Wang, Qiancheng Wang, Qihao Zhu, Qingyang Li, Qinyu Chen, Qiushi Du, Ruiling Xu, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runqiu Yin, Runxin Xu, Ruomeng Shen, Ruoyu Zhang, S. H. Liu, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaofei Cai, Shaoyuan Chen, Shengding Hu, Shengyu Liu, Shiqiang Hu, Shirong Ma, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, Songyang Zhou, Tao Ni, Tao Yun, Tian Pei, Tian Ye, Tianyuan Yue, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjie Pang, Wenjing Luo, Wenjun Gao, Wentao Zhang, Xi Gao, Xiangwen Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaokang Zhang, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xingyou Li, Xinyu Yang, Xinyuan Li, Xu Chen, Xuecheng Su, Xuehai Pan, Xuheng Lin, Xuwei Fu, Y. Q. Wang, Yang Zhang, Yanhong Xu, Yanru Ma, Yao Li, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Qian, Yi Yu, Yichao Zhang, Yifan Ding, Yifan Shi, Yiliang Xiong, Ying He, Ying Zhou, Yinmin Zhong, Yishi Piao, Yisong Wang, Yixiao Chen, Yixuan Tan, Yixuan Wei, Yiyang Ma, Yiyuan Liu, Yonglun Yang, Yongqiang Guo, Yongtong Wu, Yu Wu, Yuan Cheng, Yuan Ou, Yuanfan Xu, Yuduan Wang, Yue Gong, Yuhan Wu, Yuheng Zou, Yukun Li, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehua Zhao, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhixian Huang, Zhiyu Wu, Zhuoshu Li, Zhuping Zhang, Zian Xu, Zihao Wang, Zihui Gu, Zijia Zhu, Zilin Li, Zipeng Zhang, Ziwei Xie, Ziyi Gao, Zizheng Pan, Zongqing Yao, Bei Feng, Hui Li, J. L. Cai, Jiaqi Ni, Lei Xu, Meng Li, Ning Tian, R. J. Chen, R. L. Jin, S. S. Li, Shuang Zhou, Tianyu Sun, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xinnan Song, Xinyi Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, Dongjie Ji, Jian Liang, Jianzhong Guo, Jin Chen, Leyi Xia, Miaojun Wang, Mingming Li, Peng Zhang, Ruyi Chen, Shangmian Sun, Shaoqing Wu, Shengfeng Ye, T. Wang, W. L. Xiao, Wei An, Xianzu Wang, Xiaowen Sun, Xiaoxiang Wang, Ying Tang, Yukun Zha, Zekai Zhang, Zhe Ju, Zhen Zhang, and Zihua Qu. Deepseek-v3.2: Pushing the frontier of open large language models, 2025. URL https://arxiv.org/abs/2512.02556.

B. Efron. Bootstrap Methods: Another Look at the Jackknife. *The Annals of Statistics*, 7(1): 1 – 26, 1979. doi: 10.1214/aos/1176344552. URL https://doi.org/10.1214/aos/1176344552.

Chenrui Fan, Yize Cheng, Ming Li, Yongyuan Liang, Tianyi Zhou, and Soheil Feizi. Thinking hard, not smart: Reasoning models fail to ration test-time compute across questions, 2026. URL https://arxiv.org/abs/2608.07968.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. Thinkless: LLM learns when to think. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=ariVQf0KZx.

Nigel Fernandez, Branislav Kveton, Ryan A. Rossi, Andrew Lan, and Zichao Wang. RADAR: Reasoning-ability and difficulty-aware routing for reasoning LLMs. In *The Fourteenth Interna-*

tional Conference on Learning Representations, 2026. URL https://openreview.net/forum?id=CB6Ds5T4ae.

GLM-5-Team, :, Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, Chenzheng Zhu, Congfeng Yin, Cunxiang Wang, Gengzheng Pan, Hao Zeng, Haoke Zhang, Haoran Wang, Huilong Chen, Jiajie Zhang, Jian Jiao, Jiaqi Guo, Jingsen Wang, Jingzhao Du, Jinzhu Wu, Kedong Wang, Lei Li, Lin Fan, Lucen Zhong, Mingdao Liu, Mingming Zhao, Pengfan Du, Qian Dong, Rui Lu, Shuang-Li, Shulin Cao, Song Liu, Ting Jiang, Xiaodong Chen, Xiaohan Zhang, Xuancheng Huang, Xuezhen Dong, Yabo Xu, Yao Wei, Yifan An, Yilin Niu, Yitong Zhu, Yuanhao Wen, Yukuo Cen, Yushi Bai, Zhongpei Qiao, Zihan Wang, Zikang Wang, Zilin Zhu, Ziqiang Liu, Zixuan Li, Bojie Wang, Bosi Wen, Can Huang, Changpeng Cai, Chao Yu, Chen Li, Chengwei Hu, Chenhui Zhang, Dan Zhang, Daoyan Lin, Dayong Yang, Di Wang, Ding Ai, Erle Zhu, Fangzhou Yi, Feiyu Chen, Guohong Wen, Hailong Sun, Haisha Zhao, Haiyi Hu, Hanchen Zhang, Hanrui Liu, Hanyu Zhang, Hao Peng, Hao Tai, Haobo Zhang, He Liu, Hongwei Wang, Hongxi Yan, Hongyu Ge, Huan Liu, Huanpeng Chu, Jia'ni Zhao, Jiachen Wang, Jiajing Zhao, Jiamin Ren, Jiapeng Wang, Jiaxin Zhang, Jiayi Gui, Jiayue Zhao, Jijie Li, Jing An, Jing Li, Jingwei Yuan, Jinhua Du, Jinxin Liu, Junkai Zhi, Junwen Duan, Kaiyue Zhou, Kangjian Wei, Ke Wang, Keyun Luo, Laiqiang Zhang, Leigang Sha, Liang Xu, Lindong Wu, Lintao Ding, Lu Chen, Minghao Li, Nianyi Lin, Pan Ta, Qiang Zou, Rongjun Song, Ruiqi Yang, Shangqing Tu, Shangtong Yang, Shaoxiang Wu, Shengyan Zhang, Shijie Li, Shuang Li, Shuyi Fan, Wei Qin, Wei Tian, Weining Zhang, Wenbo Yu, Wenjie Liang, Xiang Kuang, Xiangmeng Cheng, Xiangyang Li, Xiaoquan Yan, Xiaowei Hu, Xiaoying Ling, Xing Fan, Xingye Xia, Xinyuan Zhang, Xinze Zhang, Xirui Pan, Xu Zou, Xunkai Zhang, Yadi Liu, Yandong Wu, Yanfu Li, Yidong Wang, Yifan Zhu, Yijun Tan, Yilin Zhou, Yiming Pan, Ying Zhang, Yinpei Su, Yipeng Geng, Yong Yan, Yonglin Tan, Yuean Bi, Yuhan Shen, Yuhao Yang, Yujiang Li, Yunan Liu, Yunqing Wang, Yuntao Li, Yurong Wu, Yutao Zhang, Yuxi Duan, Yuxuan Zhang, Zezhen Liu, Zhengtao Jiang, Zhenhe Yan, Zheyu Zhang, Zhixiang Wei, Zhuo Chen, Zhuoer Feng, Zijun Yao, Ziwei Chai, Ziyuan Wang, Zuzhou Zhang, Bin Xu, Minlie Huang, Hongning Wang, Juanzi Li, Yuxiao Dong, and Jie Tang. Glm-5: from vibe coding to agentic engineering, 2026. URL https://arxiv.org/abs/2602.15763.

Shashwat Goel, Joschka Strüber, Ilze Amanda Auzina, Karuna K Chandra, Ponnurangam Kumaraguru, Douwe Kiela, Ameya Prabhu, Matthias Bethge, and Jonas Geiping. Great models think alike and this undermines AI oversight. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=3Z827FtMNe.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoy-

chev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satter-

field, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://dx.doi.org/10.1038/s41586-025-09422-z.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Chengsong Huang, Langlin Huang, Jixuan Leng, Jiacheng Liu, and Jiaxin Huang. CaTS: Calibrated test-time scaling for efficient LLM reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=jrSc4RJXy1.

Gijs Kassenaar, Zhao Yang, and Vincent François-Lavet. Learning when to think: Adaptive reasoning for test-time compute allocation, 2026. URL https://arxiv.org/abs/2608.20256.

Elliot Myunghoon Kim, Avi Garg, Kenny Peng, and Nikhil Garg. Correlated errors in large language models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=kzYq2hfyHB.

Alex Kipnis, Konstantinos Voudouris, Luca M. Schulze Buschoff, and Eric Schulz. metabench - a sparse benchmark of reasoning and knowledge in large language models. In *The Thirteenth*

International Conference on Learning Representations, 2025. URL https://openreview.net/forum?id=4T33izzFpK.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Alan Li, Yixin Liu, Arpan Sarkar, Doug Downey, and Arman Cohan. Demystifying scientific problem-solving in LLMs by probing knowledge and reasoning. In *Forty-third International Conference on Machine Learning*, 2026a. URL https://openreview.net/forum?id=jB7dvOzr1a.

Pingzhi Li, Bairu Hou, Yun Zhu, Yihao Feng, Ke Ye, Tao Lei, Zhifeng Chen, Tianlong Chen, and Xianzhi Du. Adaptive thinking: Large language models know when to think in latent space. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=2i6Rp0gCq6.

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

Ryan Liu, Jiayi Geng, Addison J. Wu, Ilia Sucholutsky, Tania Lombrozo, and Thomas L. Griffiths. Mind your step (by step): Chain-of-thought can reduce performance on tasks where thinking makes humans worse, 2025. URL https://arxiv.org/abs/2410.21333.

Frederic M. Lord and Melvin R. Novick. *Statistical Theories of Mental Test Scores*. Addison-Wesley, 1968.

Haozheng Luo, Zhuolin Jiang, Md Zahid Hasan, Yan Chen, and Soumalya Sarkar. FROST: Filtering reasoning outliers with attention for efficient reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=a9dngZLqGS.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling, 2025. URL https://arxiv.org/abs/2501.19393.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. RouteLLM: Learning to route LLMs from preference data. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=8sSqNntaMr.

OpenAI, :, Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K. Arora, Yu Bai, Bowen Baker, Haiming Bao, Boaz Barak, Ally Bennett, Tyler Bertao, Nivedita Brett, Eugene Brevdo, Greg Brockman, Sebastien Bubeck, Che Chang, Kai Chen, Mark Chen, Enoch Cheung, Aidan Clark, Dan Cook, Marat Dukhan, Casey Dvorak, Kevin Fives, Vlad Fomenko, Timur Garipov, Kristian Georgiev, Mia Glaese, Tarun Gogineni, Adam Goucher, Lukas Gross, Katia Gil Guzman, John Hallman, Jackie Hehir, Johannes Heidecke, Alec Helyar, Haitang Hu, Romain Huet, Jacob Huh, Saachi Jain, Zach Johnson, Chris Koch, Irina Kofman, Dominik Kundel, Jason Kwon, Volodymyr Kyrylov, Elaine Ya Le, Guillaume Leclerc, James Park Lennon, Scott Lessans, Mario Lezcano-Casado, Yuanzhi Li, Zhuohan Li, Ji Lin, Jordan Liss, Lily, Liu, Jiancheng Liu, Kevin Lu, Chris Lu, Zoran Martinovic, Lindsay McCallum, Josh McGrath, Scott McKinney, Aidan McLaughlin, Song Mei, Steve Mostovoy, Tong Mu, Gideon Myles, Alexander Neitz, Alex Nichol, Jakub Pachocki, Alex Paino, Dana Palmie, Ashley Pantuliano, Giambattista Parascandolo, Jongsoo Park, Leher Pathak, Carolina Paz, Ludovic Peran, Dmitry Pimenov, Michelle Pokrass, Elizabeth Proehl, Huida Qiu, Gaby Raila, Filippo Raso, Hongyu Ren, Kimmy Richardson, David Robinson, Bob Rotsted, Hadi Salman, Suvansh Sanjeev, Max Schwarzer, D. Sculley, Harshit Sikchi, Kendal Simon, Karan Singhal, Yang Song, Dane Stuckey, Zhiqing Sun, Philippe Tillet, Sam Toizer, Foivos Tsimpourlas, Nikhil Vyas, Eric Wallace, Xin Wang, Miles Wang, Olivia Watkins, Kevin Weil, Amy Wendling, Kevin Whinnery,

Cedric Whitney, Hannah Wong, Lin Yang, Yu Yang, Michihiro Yasunaga, Kristen Ying, Wojciech Zaremba, Wenting Zhan, Cyril Zhang, Brian Zhang, Eddie Zhang, and Shengjia Zhao. gpt-oss-120b gpt-oss-20b model card, 2025. URL https://arxiv.org/abs/2508.10925.

OpenAI, :, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrew Duberstein, Andrew Kondrich, Andrey Mishchenko, Andy Applebaum, Angela Jiang, Ashvin Nair, Barret Zoph, Behrooz Ghorbani, Bohan Zhang, Ben Rossen, Benjamin Sokolowsky, Boaz Barak, Bob McGrew, Borys Minaiev, Botao Hao, Bowen Baker, Brandon Houghton, Brandon McKinzie, Brydon Eastman, Camillo Lugaresi, Cary Bassin, Cary Hudson, Chak Ming Li, Charles de Bourcy, Chelsea Voss, Chen Shen, Chong Zhang, Chris Koch, Chris Orsinger, Christopher Hesse, Claudia Fischer, Clive Chan, Dan Roberts, Daniel Kappler, Daniel Levy, Daniel Selsam, David Dohan, David Farhi, David Mely, David Robinson, Dimitris Tsipras, Doug Li, Dragos Oprica, Eben Freeman, Eddie Zhang, Edmund Wong, Elizabeth Proehl, Enoch Cheung, Eric Mitchell, Eric Wallace, Erik Ritter, Evan Mays, Fan Wang, Felipe Petroski Such, Filippo Raso, Florencia Leoni, Foivos Tsimpourlas, Francis Song, Fred von Lohmann, Freddie Sulit, Geoff Salmon, Giambattista Parascandolo, Gildas Chabot, Grace Zhao, Greg Brockman, Guillaume Leclerc, Hadi Salman, Haiming Bao, Hao Sheng, Hart Andrin, Hessam Bagherinezhad, Hongyu Ren, Hunter Lightman, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian Osband, Ignasi Clavera Gilaberte, Ilge Akkaya, Ilya Kostrikov, Ilya Sutskever, Irina Kofman, Jakub Pachocki, James Lennon, Jason Wei, Jean Harb, Jerry Twore, Jiacheng Feng, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joaquin Quiñonero Candela, Joe Palermo, Joel Parish, Johannes Heidecke, John Hallman, John Rizzo, Jonathan Gordon, Jonathan Uesato, Jonathan Ward, Joost Huizinga, Julie Wang, Kai Chen, Kai Xiao, Karan Singhal, Karina Nguyen, Karl Cobbe, Katy Shi, Kayla Wood, Kendra Rimbach, Keren Gu-Lemberg, Kevin Liu, Kevin Lu, Kevin Stone, Kevin Yu, Lama Ahmad, Lauren Yang, Leo Liu, Leon Maksin, Leyton Ho, Liam Fedus, Lilian Weng, Linden Li, Lindsay McCallum, Lindsey Held, Lorenz Kuhn, Lukas Kondraciuk, Lukasz Kaiser, Luke Metz, Madelaine Boyd, Maja Trebacz, Manas Joglekar, Mark Chen, Marko Tintor, Mason Meyer, Matt Jones, Matt Kaufer, Max Schwarzer, Meghan Shah, Mehmet Yatbaz, Melody Y. Guan, Mengyuan Xu, Mengyuan Yan, Mia Glaese, Mianna Chen, Michael Lampe, Michael Malek, Michele Wang, Michelle Fradin, Mike McClay, Mikhail Pavlov, Miles Wang, Mingxuan Wang, Mira Murati, Mo Bavarian, Mostafa Rohaninejad, Nat McAleese, Neil Chowdhury, Neil Chowdhury, Nick Ryder, Nikolas Tezak, Noam Brown, Ofir Nachum, Oleg Boiko, Oleg Murk, Olivia Watkins, Patrick Chao, Paul Ashbourne, Pavel Izmailov, Peter Zhokhov, Rachel Dias, Rahul Arora, Randall Lin, Rapha Gontijo Lopes, Raz Gaon, Reah Miyara, Reimar Leike, Renny Hwang, Rhythm Garg, Robin Brown, Roshan James, Rui Shu, Ryan Cheu, Ryan Greene, Saachi Jain, Sam Altman, Sam Toizer, Sam Toyer, Samuel Miserendino, Sandhini Agarwal, Santiago Hernandez, Sasha Baker, Scott McKinney, Scottie Yan, Shengjia Zhao, Shengli Hu, Shibani Santurkar, Shraman Ray Chaudhuri, Shuyuan Zhang, Siyuan Fu, Spencer Papay, Steph Lin, Suchir Balaji, Suvansh Sanjeev, Szymon Sidor, Tal Broda, Aidan Clark, Tao Wang, Taylor Gordon, Ted Sanders, Tejal Patwardhan, Thibault Sottiaux, Thomas Degry, Thomas Dimson, Tianhao Zheng, Timur Garipov, Tom Stasi, Trapit Bansal, Trevor Creech, Troy Peterson, Tyna Eloundou, Valerie Qi, Vineet Kosaraju, Vinnie Monaco, Vitchyr Pong, Vlad Fomenko, Weiyi Zheng, Wenda Zhou, Wenting Zhan, Wes McCabe, Wojciech Zaremba, Yann Dubois, Yinghai Lu, Yining Chen, Young Cha, Yu Bai, Yuchen He, Yuchen Zhang, Yunyun Wang, Zheng Shao, and Zhuohan Li. Openai o1 system card, 2026. URL https://arxiv.org/abs/2412.16720.

Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. tinybenchmarks: evaluating LLMs with fewer examples. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=qAml3FpfhG.

Akshara Prabhakar, Thomas L. Griffiths, and R. Thomas McCoy. Deciphering the factors influencing the efficacy of chain-of-thought: Probability, memorization, and noisy reasoning, 2024. URL https://arxiv.org/abs/2407.01687.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof qa benchmark, 2023. URL https://arxiv.org/abs/2311.12022.

Yangjun Ruan, Chris J. Maddison, and Tatsunori Hashimoto. Observational scaling laws and the predictability of langauge model performance. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=On5WIN7xyD.

Parshin Shojaee, Seyed Iman Mirzadeh, Keivan Alizadeh, Maxwell Horton, Samy Bengio, and Mehrdad Farajtabar. The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=YghiOusmvw.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters, 2024. URL https://arxiv.org/abs/2408.03314.

Wei Song, Zhenya Huang, Cheng Cheng, Weibo Gao, Bihan Xu, GuanHao Zhao, Fei Wang, and Runze Wu. Irt-router: Effective and interpretable multi-llm routing via item response theory, 2025. URL https://arxiv.org/abs/2506.01048.

Charles Spearman. Correlation calculated from faulty data. *British Journal of Psychology*, 3:271–295, 1910.

Zayne Rea Sprague, Fangcong Yin, Juan Diego Rodriguez, Dongwei Jiang, Manya Wadhwa, Prasann Singhal, Xinyu Zhao, Xi Ye, Kyle Mahowald, and Greg Durrett. To cot or not to cot? chain-of-thought helps mainly on math and symbolic reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=w6nlcS8Kkn.

Jinyan Su, Jennifer Healey, Preslav Nakov, and Claire Cardie. Between underthinking and overthinking: An empirical study of reasoning length and correctness in llms, 2025. URL https://arxiv.org/abs/2505.00127.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, Hanjie Chen, and Xia Hu. Stop overthinking: A survey on efficient reasoning for large language models, 2025. URL https://arxiv.org/abs/2503.16419.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, Mayank Chaturvedi, Aditya Chawla, Victor Cotruta, Alice Coucke, Phil Culliton, Robert Dadashi, Lucas Dixon, Mohamed Elhawaty, Utku Evci, Clément Farabet, Johan Ferret, Filippo Galgani, Sertan Girgin, Jean-Bastien Grill, Maarten Grootendorst, Jiaxian Guo, Cassidy Hardin, Yanzhang He, Steven M. Hernandez, Omri Homburger, Léonard Hussenot, Juyeong Ji, Armand Joulin, Aishwarya Kamath, Parnian Kassraie, Olivier Lacombe, Preethi Lahoti, Gaël Liu, Gus Martins, Luciano Martins, Tatiana Matejovicova, Ramona Merhej, Nikola Momchev, Sneha Mondal, Ryan Mullins, Sindhu Raghuram Panyam, Shreya Pathak, Sarah Perrin, André Susano Pinto, Etienne Pot, Angéline Pouget, Alexandre Ramé, Sabela Ramos, Douglas Reid, David Rim, Morgane Rivière, Karsten Roth, Louis Rouillard, Omar Sanseviero, Pier Giuseppe Sessa, Shane Settle, Danila Sinopalnikov, Sara Smoot, Piotr Stanczyk, Andreas Steiner, Lawrence Stewart, Ilya Tolstikhin, Michael Tschannen, Anton Tsitsulin, Nino Vieillard, Renjie Wu, Pingmei Xu, Haichuan Yang, Edouard Yvinec, Biao Zhang, Li Zhang, Joe Zou, Nicolas Aagnes, Abdelrahman Abdelhamed, Jakub Adamek, Shivani Agrawal, Shubham Agrawal, Ibrahim Alabdulmohsin, Jean Baptiste Alayrac, Uri Alon, Chandramouli Amarnath, Ankesh Anand, Chrysovalantis Anastasiou, Setareh Ariafar, François-Xavier Aubet, Kyriakos Axiotis, Federico Barbero, Joelle Barral, Alexei Bendebury, Urs Bergmann, Stanley Bileschi, Kat Black, Mathieu Blondel, Sebastian Borgeaud, Arthur Bražinskas, Ryan Burnell, Robert Busa-Fekete, Mu Cai, Daniele Calandriello, Glenn Cameron, Charlotte Caucheteux, Rahma Chaabouni, Garima Chadha, Jetha Chan, Blake Jianhang Chen, Jesse Chen, Lin Chen, Xu Chen, Derek Cheng, Tzu hsiang Chien, Nikolai Chinaev, Yi Chou, Zhaohui Chu, Benjamin Coleman, Pooja Consul, Sam Conway-Rahman, Scott Crowell, Dylan Cutler, Vivek Dani, Samira Daruki, Anil Das, Daniel Deutsch, Nishanth Dikkala, Li Ding, Qiuhan Ding, Shenil Dodhia, Konstantin Donhauser, Tulsee Doshi, Anca Dragan, Alex Druinsky, Sahil Dua, Zoltan Egyed,

Danielle Eisenbud, Daniel Eppens, Cindy Fan, Bahare Fatemi, Yassir Fathullah, Vlad Feinberg, Milen Ferev, Sebastian Flennerhag, Takumi Fujimoto, João Gabriel Oliveira, Isaac Galatzer-Levy, João Gante, Simon Geisler, Soham Ghosal, Antonious M. Girgis, Tamara von Glehn, Alec Go, Alhaad Gokhale, Alex Grills, Yiming Gu, Mayank Gupta, Pramod Gupta, Guru Guruganesh, Raia Hadsell, Hamza Harkous, Jitendra Harlalka, Demis Hassabis, Anja Hauth, Joe Heyward, Arian Hosseini, Chih-Yang Hsia, I-Hung Hsu, Xiaopeng Huang, Yangsibo Huang, Kevin Hui, Adrian Hutter, Te I, Fotis Iliopoulos, Advait Jain, Ganesh Jawahar, Ziwei Ji, Qilin Jin, Melvin Johnson, Kandarp Joshi, Arun Kandoor, Wang-Cheng Kang, Koray Kavukcuoglu, Mehran Kazemi, Kathleen Kenealy, Amr Khalifa, Phoebe Kirk, Ivan Korotkov, Suraj Kothawade, Vitaly Kovalev, Neel Kovelamudi, Adam Kraft, Ravin Kumar, Vivek Kumar, Harish Kuppam, Justin Lannin, Chen-Yu Lee, Seungji Lee, Dmitry Lepikhin, Alon Levkovitch, Dongdong Li, Qiujia Li, Valentin Liévin, Ethan Lin, Ziqian Lin, Casper Liu, Tianlin Liu, Tianqi Liu, Xin Liu, Ivan Lobov, Mayank Lunayach, Min Ma, Gagan Madan, Andrii Maksai, Eric Malmi, Michal Matuszak, Daniel McDuff, Gaurav Menghani, Maciej Mikuła, Daniil Mirylenka, Karolis Misiunas, Vedant Misra, Andreea Mitran, Kareem Mohamed, Maksim Mukha, Eric Noland, James O'Donnell, Brendan O'Donoghue, Kate Olszewska, Bernett Orlando, Wanqiong Pan, Rina Panigrahy, Unnati Parekh, Nicolas Perez-Nieves, Chunjong Park, Eric Paskie, Liqian Peng, Bryce Petrini, Slav Petrov, Jonas Pfeiffer, Bilal Piot, Martyna Plomecka, Siim Poder, Octavio Ponce, Arijit Pramanik, David Racz, Anish Rajan, Michelle Ramanovich, Anand Rao, Marvin Ritter, Vitor Rodrigues, Evan Rosen, Mikołaj Rybiński, Noveen Sachdeva, Michaël E. Sander, Rohit Sathyanarayana, Sagar Savla, Samuel Schmidgall, Tal Schuster, George Scrivener, Benoit Seguin, Andrew Sellergren, Aliaksei Severyn, Izhak Shafran, Dhruv Shah, Bobak Shahriari, Yuan Shangguan, Ashish Shenoy, Pradeep Shenoy, Rakesh Shivanna, Pauline Sho, Lucas Spangher, Wojciech Stokowiec, Tim Strother, Yao Su, Yinghao Sun, Mukund Sundararajan, Andrea Tacchetti, Mor Hazan Taege, Pouya Tafti, Jean Tarbouriech, Chetan Tekur, Shantanu Thakoor, Rahul Thapa, Madeleine Traverse, Lenart Treven, Tao Tu, Chien Te Tung, Çağlar Ünlü, Petar Veličković, Malini Pooni Venkat, Sagar Gubbi Venkatesh, Vidya Venkiteswaran, Francesco Visin, Alex Vitvitskyi, Kiran Vodrahalli, Weiyi Wang, Xin Wang, Tris Warkentin, Jan Wassenberg, John Wieting, Cindy Wu, Lechao Xiao, Hao Xu, Yuhui Xu, Fuzhao Xue, Arun Yadav, Jun Yan, Antoine Yang, Lin Yang, Ming-Hsuan Yang, Ziyu Ying, Jae Hyeon Yoo, Morteza Zadimoghaddam, Sajjad Zafar, Fred Zhang, Jiageng Zhang, Jianyi Zhang, Xiaofan Zhang, Chao Zhao, David Zhou, and Chen Zou. Gemma 4 technical report, 2026a. URL https://arxiv.org/abs/2607.02770.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, S. H. Cai, Yuan Cao, Ziwei Chai, Y. Charles, H. S. Che, Cheng Chen, Guanduo Chen, Huarong Chen, Jia Chen, Jianlong Chen, Jun Chen, Kefan Chen, Liang Chen, Ruijue Chen, Xinhao Chen, Yanru Chen, Yanxu Chen, Yicun Chen, Yimin Chen, Yingjiang Chen, Yuankun Chen, Yujie Chen, Yutian Chen, Zhirong Chen, Ziwei Chen, Dazhi Cheng, Yean Cheng, Minghan Chu, Jialei Cui, Jiaqi Deng, Muxi Diao, Hao Ding, Mengfan Dong, Mengnan Dong, Yuxin Dong, Yuhao Dong, Angang Du, Chenzhuang Du, Dikang Du, Lingxiao Du, Yulun Du, Yu Fan, Shengjun Fang, Qiulin Feng, Yichen Feng, Garimugai Fu, Kelin Fu, Hongcheng Gao, Tong Gao, Yuyao Ge, Shangyi Geng, Chengyang Gong, Xiaochen Gong, Zhuoma Gongque, Qizheng Gu, Xinran Gu, Yicheng Gu, Longyu Guan, Shuhao Guan, Yuanying Guo, Xiaoru Hao, Dailan He, Tianhong He, Weiran He, Wenyang He, Yibo He, Yunjia He, Chao Hong, Hao Hu, Jiaxi Hu, Yangyang Hu, Zhenxing Hu, Ke Huang, Ruiyuan Huang, Weixiao Huang, Zhiqi Huang, Chaobo Jia, Tao Jiang, Zhejun Jiang, Xinyi Jin, Yu Jing, Guokun Lai, Aidi Li, C. Li, Cheng Li, Fang Li, Guanghe Li, Guanyu Li, Haitao Li, Haoyang Li, Jia Li, Jingwei Li, Junxiong Li, Lincan Li, Mo Li, Weihong Li, Wentao Li, Xinhang Li, Xinhao Li, Yang Li, Yanhao Li, Yiwei Li, Yuxiao Li, Zhaowei Li, Zhaoxi Li, Zheming Li, Weilong Liao, Jiawei Lin, Xiaohan Lin, Yibo Lin, Zhishan Lin, Zichao Lin, Cheng Liu, Chenyu Liu, Hongzhang Liu, Liang Liu, Shaowei Liu, Shudong Liu, Shuran Liu, Tianwei Liu, Tianyu Liu, Weizhou Liu, Xiangyan Liu, Yangyang Liu, Yanming Liu, Yibo Liu, Yuanxin Liu, Zhengying Liu, Zhongnuo Liu, Enzhe Lu, Haoyu Lu, Zhiyuan Lu, G. Luo, Junyu Luo, Tongxu Luo, Yashuo Luo, Long Ma, Shaoguang Mao, Yuan Mei, Xin Men, Fanqing Meng, Zhiyong Meng, Yibo Miao, Minqing Ni, Kun Ouyang, Siyuan Pan, Bo Pang, Yuchao Qian, Ruoyu Qin, Zeyu Qin, Jiezhong Qiu, Bowen Qu, Zeyu Shang, Youbo Shao, Tianxiao Shen, Zhennan Shen, Juanfeng Shi, Lidong Shi, Shengyuan Shi, Feifan Song, Pengwei Song, Tianhui Song, Xiaoxi Song, Hongjin Su, Jianlin Su, Zhaochen Su, Lin Sui, Jinsong Sun, Junyao Sun, Tongyu Sun, Flood Sung, Yunpeng Tai, Chuning Tang, Heyi Tang, Xiaojuan Tang, Zhengyang Tang, Jiawen Tao, Shiyuan Teng, Chaoran Tian, Pengfei Tian, Bowen Wang, Chensi Wang, Chuang Wang, Congcong Wang, Dingkun Wang,

Dinglu Wang, Dongliang Wang, Feng Wang, Hailong Wang, Haiming Wang, Hao Wang, Hengzhi Wang, Huaqing Wang, Hui Wang, Jiahao Wang, Jinhong Wang, Jiuzheng Wang, Kaixin Wang, Linian Wang, Qibin Wang, Shengjie Wang, Shuyi Wang, Si Wang, Wei Wang, Xiaochen Wang, Xinyuan Wang, Yao Wang, Yejie Wang, Yipu Wang, Yiqin Wang, Yucheng Wang, Yuzhi Wang, Zhaoji Wang, Zhaowei Wang, Zhengtao Wang, Zhexu Wang, Zifan Wang, Zihan Wang, Zizhe Wang, Chu Wei, Ming Wei, Chuan Wen, Zichen Wen, Chengjie Wu, Haoning Wu, Junyan Wu, Rucong Wu, Wenhao Wu, Yuefeng Wu, Yuhao Wu, Yuxin Wu, Zijian Wu, Chenjun Xiao, Jin Xie, Xiaotong Xie, Yuchong Xie, Bowei Xing, Boyu Xu, Jianfan Xu, Jing Xu, Jinjing Xu, L. H. Xu, Lin Xu, Suting Xu, Weixin Xu, Xinbo Xu, Xinran Xu, Yangchuan Xu, Yichang Xu, Yuemeng Xu, Zelai Xu, Ziyao Xu, Junjie Yan, Yuzi Yan, Guangyao Yang, Hao Yang, Junwei Yang, Kai Yang, Ningyuan Yang, Xiaofei Yang, Xinlong Yang, Xinyu Yang, Ying Yang, Yi Yang, Yi Yang, Zhen Yang, Zhilin Yang, Zonghan Yang, Haotian Yao, Dan Ye, Haoran Ye, Wenjie Ye, Zhuorui Ye, Peng Yebo, Bohong Yin, Chengzhen Yu, Longhui Yu, Tao Yu, Tianxiang Yu, Enming Yuan, Mengjie Yuan, Xiaokun Yuan, Yang Yue, Weihao Zeng, Dunyuan Zha, Haobing Zhan, Dehao Zhang, Hao Zhang, Jin Zhang, Puqi Zhang, Qiao Zhang, Rui Zhang, Xiaobin Zhang, Xiaoyun Zhang, Y. Zhang, Yadong Zhang, Yangkun Zhang, Yichi Zhang, Yizhi Zhang, Yongting Zhang, Yu Zhang, Yushun Zhang, Yutao Zhang, Yutong Zhang, Zheng Zhang, Chenguang Zhao, Feifan Zhao, Jinxiang Zhao, Shuai Zhao, Xiangyu Zhao, Xuanle Zhao, Yikai Zhao, Zijia Zhao, Huabin Zheng, Ruihan Zheng, Shaojie Zheng, Tengyang Zheng, Junfeng Zhong, Longguang Zhong, Weiming Zhong, M. Zhou, Runjie Zhou, Xinyu Zhou, Zaida Zhou, Jinguo Zhu, Liya Zhu, Xinhao Zhu, Yuxuan Zhu, Zhen Zhu, Jingze Zhuang, Weiyu Zhuang, Ying Zou, and Xinxing Zu. Kimi k2.5: Visual agentic intelligence, 2026b. URL https://arxiv.org/abs/2602.02276.

Songjun Tu, Jiahao Lin, Qichao Zhang, Xiangyu Tian, Linjing Li, Xiangyuan Lan, and Dongbin Zhao. Learning when to think: Shaping adaptive reasoning in r1-style models via multi-stage RL. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=Hs3FrjwyVZ.

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

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=y10DM6R2r3.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for LLM problem-solving. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=VNckp7JEHn.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Zhiyuan Zhai, Bingcong Li, Bingnan Xiao, Ming Li, and Xin Wang. Adaptive test-time compute allocation for reasoning llms via constrained policy optimization, 2026. URL https://arxiv.org/abs/2604.14853.

Jiajie Zhang, Nianyi Lin, Lei Hou, Ling Feng, and Juanzi Li. Adaptthink: Reasoning models can learn when to think, 2025. URL https://arxiv.org/abs/2505.13417.

Shu Zhou, Rui Ling, Junan Chen, Xin Wang, Tao Fan, and Hao Wang. When more thinking hurts: Overthinking in llm test-time compute scaling, 2026. URL https://arxiv.org/abs/2604.10739.

Jason Zhu and Hongyu Li. Towards concise and adaptive thinking in large reasoning models: A survey, 2025. URL https://arxiv.org/abs/2507.09662.

A Detailed Protocol and Audit

A.1 Data inclusion rules

The analysis reads data from the experiment archive. It retains zero-shot, few-shot, and CoT strategies because they exist in both modes. DSPy is removed from paired comparisons because it was run only in non-thinking mode. For GPT-OSS, low effort is non-thinking and high effort is thinking; medium effort is excluded. For GPT-OSS, "non-thinking" denotes low reasoning effort, not an absence of reasoning. Both GPT-OSS models are retained in the request-matched analysis and are assigned to the same developer. Raw calls are averaged within model-question-strategy-mode, then prompts are averaged within model-question-mode. The primary analysis removes no row because of correctness, response length, extraction validity, or finish reason; the clean-item sensitivity instead removes the entire question if any included response carries an invalid-extraction or truncation flag.

A.2 Question selection and answer sets

The MMLU-Pro loader restricts the public test split to six STEM categories and uses Python's pseudorandom sampler with seed 42 to draw 50 items from each category. Questions were selected before model evaluation, without using model outputs. The source STEM pool contains roughly 5,878 questions. The evaluation sample contains 265 ten-option items and 35 items with three to nine options. GPQA-Diamond uses all 198 items, each with four options. Option counts are invariant across modes and models; a dedicated MMLU-Pro sensitivity adds option-count indicators to the category-specific quartic difficulty baseline.

A.3 Model roster and developer mapping

The 18 paired models are DeepSeek-V3.2; Gemma-4-26B-A4B-IT; Gemma-4-31B-IT; GLM-5; GPT-OSS-20B and 120B; Kimi-K2.5; Mistral-Small-4-119B; Qwen3-8B, 14B, 30B-A3B, 32B, and 235B-A22B; and Qwen3.5-9B, 27B, 35B-A3B, 122B-A10B, and 397B-A17B. Qwen variants map to Alibaba, Gemma to Google, GPT-OSS to OpenAI, GLM to Zhipu, Kimi to Moonshot, and the remaining named families to their developers. The independent difficulty panel contains Llama-3.1-70B-Instruct, Llama-3.2-1B-Instruct, Llama-3.3-70B-Instruct, Llama-4-Maverick, and Llama-4-Scout (Grattafiori et al., 2024); Ministral-3 at 3B, 8B, and 14B; Mistral-Large-3-675B; and Phi-4 (Abdin et al., 2024); these ten models contribute only non-thinking correctness. The developer-disjoint panel excludes all four Mistral-family models, leaving five Meta models and Phi-4 from Microsoft.

A.4 Developer split construction

For each question and developer, model-level gains are averaged so Alibaba's ten paired variants receive the same total weight as a one-model developer. Each of the 35 splits selects three developers as group A and uses the complementary four as group B. Raw Pearson and Spearman correlations are

computed across questions. Category adjustment subtracts fitted category means. Flexible adjustment fits a separate intercept and quartic polynomial of the group's non-thinking solve rate within each category, allowing the fitted difficulty relationship to vary by subject.

A.5 Leave-one-developer-out construction

For a held-out developer, the item predictor is the equally weighted mean gain for the same question across the six training developers. The simple baseline is the training mean by category. The flexible baseline is fit on training-developer question means using category-specific quartic functions of training non-thinking solve rate. The adjusted correlation compares actual and predicted residuals relative to that baseline; MSE comparisons use the unresidualized held-out gain.

A.6 Cluster bootstrap and permutation

The pooled adjusted statistic contains seven held-out predictions for each question. Bootstrap replicates resample question IDs with replacement and retain each sampled question's full vector of held-out-developer observations; residuals are computed once from the training-fitted baselines and are not refit per replicate, so the intervals are conditional on those fits. Because whole questions are resampled, this is cluster-robust inference at the question level, allowing arbitrary correlation within a question across developers, prompts, and modes (Colin Cameron & Miller, 2015). The developer dimension contains only seven clusters, so we do not cluster over developers; instead, inference is conditional on the roster, whose influence Section 5 probes directly. For the permutation test, we shuffle item-level residual predictions within strata defined by held-out developer, category, and baseline-prediction quintile. The main analysis defines quintiles using each question's mean baseline prediction across the seven folds. Although each fold excludes its target developer when fitting its baseline, this cross-fold mean includes folds in which that developer is a donor. A check using fold-specific, donor-only strata gives the same minimum attainable value of $(1+k)/(1+1000) = 0.001$ with zero exceedances. Given the calibration in Appendix A.18, these permutation outputs are uncalibrated diagnostics rather than significance levels.

A.7 GPQA-Diamond replication

The replication reuses the exact paired-mode, three-prompt, developer-equal aggregation and validation code on 198 GPQA-Diamond items. We apply the analysis to all 198 GPQA-Diamond questions and the same 18 paired models; Table A1 reports transfer to each held-out developer. The flexible control uses GPQA broad subdomain and a subdomain-specific quartic of training non-thinking solve rate. The bootstrap and conditional permutation are repeated with GPQA question IDs and the same baseline-prediction quintiles.

Table A1: GPQA-Diamond transfer to each held-out developer.

Developer	Raw r	Flexible r	IRT r	Δ_Q
Alibaba	0.495	0.433	0.393	0.333
DeepSeek	0.261	0.262	0.245	0.192
Google	0.316	0.219	0.212	0.186
Mistral	0.335	0.323	0.315	0.324
Moonshot	0.193	0.166	0.142	0.127
OpenAI	0.400	0.329	0.317	0.280
Zhipu	0.510	0.426	0.431	0.500
Median	0.335	0.323	0.315	0.280

A.8 IRT-style control

For LODO evaluation, the control is estimated using the six donor developers. For split correlations, controls are fitted separately within the two developer groups. A respondent is a model–prompt pair, its empirical ability is the logit of overall non-thinking accuracy with clipping for numerical stability, and each item receives regularized logistic intercept and slope estimates (ridge 1.0 on slopes and 0.01 on intercepts, Newton updates clipped to ± 1, at most 30 iterations, accuracies clipped to

$[10^{-3}, 1-10^{-3}]$). The downstream control augments the category-specific quartic solve-rate design with standardized item discrimination and its square; difficulty is represented by the quartic solve-rate terms, and fitted 2PL difficulty is not included separately. Explicitly, the fitted baseline for question q with category $c(q)$, training non-thinking solve rate $\widehat{p}_q$, and standardized discrimination $\widehat{a}_q$ is

$$\widehat{g}(q) \;=\; f_{c(q)}\big(\widehat{p}_q\big) \;+\; \gamma_1 \widehat{a}_q \;+\; \gamma_2 \widehat{a}_q^2, \tag{3}$$

where each f_c is a quartic polynomial with category-specific coefficients and all coefficients are estimated jointly by least squares on training-developer question means. Adjusted correlations subtract $\widehat{g}(q)$ from both the item predictor and the held-out gain. The logistic fits permit negative discrimination slopes. We use "IRT-style" because ability is estimated first and then held fixed. This analysis tests whether transferred gains add predictive information after augmenting the difficulty control with item discrimination.

A.9 Independent difficulty and request-matched transfer

For each benchmark, the independent difficulty score averages non-thinking correctness first within each of the ten reserved models and then equally across models. Reliability is estimated by enumerating all 126 unique five-versus-five model partitions, correlating their question solve rates, taking the median, and applying the Spearman–Brown formula (Spearman, 1910; Brown, 1910); the resulting full-panel reliability is 0.912 on MMLU-Pro and 0.838 on GPQA-Diamond. In every developer split and LODO fold, this external score replaces paired-model non-thinking solve rate inside the same category-specific quartic design; it is never estimated from either side's paired gain. The developer-disjoint analysis repeats the procedure with the six Meta and Microsoft models. Request-matched analyses restrict the paired set to seven models from five developers and enumerate all ten two-versus-three developer splits; Table A2 reports the results. The panel is family-clustered, so many five-versus-five splits place related models in both halves; splitting instead by family (the five Meta models versus the Mistral and Microsoft models) gives half-panel correlations of 0.79 and 0.73, implying full-panel reliabilities of 0.88 and 0.84, similar to and on MMLU-Pro slightly below the 126-split estimates. Two roster sensitivities give the same transfer conclusion: excluding the GPT-OSS developer leaves ten three-versus-three partitions with median adjusted r of 0.59 and 0.50 and pooled LODO r of 0.28 and 0.27, and sampling one variant per developer (all 40 rosters) gives median pooled LODO r of 0.26 and 0.25 (minima 0.22 and 0.20).

Table A2: Transfer under independently measured difficulty and matched logged request fields. CIs are from the question-cluster bootstrap. Appendix A.18 reports the permutation calibration.

Benchmark	Sensitivity	Median split r	Pooled LODO r [95% CI]
MMLU-Pro	External panel, 10 models	0.549	0.316 [0.248,0.374]
MMLU-Pro	Developer-disjoint panel, 6 models	0.567	0.329 [0.264,0.388]
MMLU-Pro	Request-matched, 7 models/5 devs.	0.526	0.255 [0.170,0.330]
GPQA-Diamond	External panel, 10 models	0.456	0.287 [0.194,0.361]
GPQA-Diamond	Developer-disjoint panel, 6 models	0.465	0.295 [0.212,0.368]
GPQA-Diamond	Request-matched, 7 models/5 devs.	0.504	0.268 [0.152,0.361]

A.10 Extraction, truncation, and option-count sensitivity

The primary analysis treats missing or unparseable extracted answers as incorrect and retains truncated responses. Invalid-extraction and truncation flags are supplied by the archived pipeline, independently of the transfer analysis. Invalid-extraction rates are 0.006% versus 0.246% for MMLU-Pro non-thinking and thinking responses and 0% versus 0.327% for GPQA-Diamond; truncation rates are 0.031% versus 0.123% and 0.178% versus 0.290%. A conservative item-level sensitivity removes every question with at least one flag in any included response, excluding 33 questions on each benchmark. The MMLU-Pro option-count sensitivity instead retains all items and adds indicators for three through ten answer options to the flexible baseline. Table A3 reports both sensitivities. Removing the five most positive and five most negative pooled-gain questions leaves median adjusted split correlations of 0.49 on MMLU-Pro and 0.38 on GPQA-Diamond, with pooled LODO correlations of 0.21 and 0.14. Transfer remains positive after this exclusion, although its magnitude decreases, particularly on GPQA-Diamond.

Table A3: Sensitivity to flagged responses and MMLU-Pro answer-option counts. CIs are from the question-cluster bootstrap. Appendix A.18 reports the permutation calibration.

Benchmark	Sensitivity	Items	Median split r	Pooled LODO r [95% CI]
MMLU-Pro	Exclude any invalid/truncated item	267	0.586	0.273 [0.175,0.361]
MMLU-Pro	Add option-count indicators	300	0.600	0.298 [0.221,0.365]
GPQA-Diamond	Exclude any invalid/truncated item	165	0.569	0.309 [0.187,0.411]

A.11 Random-effects reference correlations

A held-out developer's observed gains combine shared question effects, developer-specific structure, and sampling variation. An exchangeable-developer random-effects model gives reference correlations of 0.522 on MMLU-Pro (question-bootstrap interval $[0.443, 0.588]$) and 0.549 on GPQA-Diamond ($[0.455, 0.627]$) between a six-developer mean and one developer after flexible adjustment. These references mix stable developer-specific variation with sampling noise and use a different residualization from the pooled LODO statistic; they provide context for the observed correlations, but are not exact noise ceilings. For each benchmark we fit a balanced two-way method-of-moments decomposition to developer-by-question gains, both raw and after flexible adjustment. If σ_q^2 is shared question variance and σ_{dq}^2 is developer-by-question variance, the reliability of a k-developer mean is $\sigma_q^2/(\sigma_q^2 + \sigma_{dq}^2/k)$. The six-to-one reference correlation is the square root of the product of the six-developer and single-developer reliabilities. Question bootstrap intervals refit the variance components in every replicate.

Table A4: Random-effects reference correlations under the exchangeable-developer model.

Benchmark	Scale	Rel., 1 dev.	Rel., 6 dev.	Reference r, 6-to-1 [95% CI]
MMLU-Pro	Raw	0.278	0.698	0.441 [0.391,0.490]
MMLU-Pro	Adjusted	0.355	0.768	0.522 [0.443,0.588]
GPQA-Diamond	Raw	0.202	0.603	0.349 [0.263,0.423]
GPQA-Diamond	Adjusted	0.383	0.788	0.549 [0.455,0.627]

A.12 Repeat-based reliability calibration from AIME

AIME is the only benchmark in the archive with repeated sampling: five runs per model, question, prompt, and mode, over 90 questions and the same 18 paired models. Balanced two-versus-two split halves over four runs give the reliability of a two-run gain estimate; the Spearman–Brown formula (Spearman, 1910; Brown, 1910) converts reliability between run counts, and the five-run two-way decomposition supplies the shared-question and developer-by-question variances. Solving the two relations separates three components of a single-call developer-level item gain, the protocol unit used on MMLU-Pro and GPQA-Diamond. Explicitly, the centered two-run split-half correlation satisfies $R_2 = A/(A + N_1/2)$ with $A = S + I$, and the five-run decomposition gives $S = \sigma_q^2$ and $\sigma_e^2 = I + N_1/5$, where S, I, and N_1 are the shared-question, developer-specific, and single-call noise variances. Because the two-way components remove developer mean effects, the split-half correlation entering these relations is centered within developer before correlating; the uncentered stacked correlation, which also contains stable developer offsets, is reported for reference only.

Table A5: Repeat-based measurement calibration on AIME (90 items, 18 models, 7 developers, 5 runs).

Quantity	Value
Split-half r, two-run developer-level gains (stacked, incl. offsets)	0.853
Split-half r, centered within developer (used in decomposition)	0.793
Split-half r, two-run pooled 7-developer $\widehat{\tau}_q$	0.943
Single-developer reliability, five-run (two-way model)	0.374
Single-call variance share: shared question signal	0.271
Single-call variance share: developer-specific structure	0.385
Single-call variance share: sampling noise	0.343
Sampling-noise share of interaction-plus-noise bucket	0.471
Implied single-call six-to-one reference correlation	0.433

Per-developer split-half reliabilities range from 0.456 (Google, two model variants) to 0.962 (Alibaba, ten variants), with the remaining developers between 0.69 and 0.90. The moment-based

decomposition assumes exchangeable developers and homogeneous sampling noise, although variant counts differ. Because AIME also differs from the primary benchmarks in format and truncation rate, its estimates provide a benchmark-specific comparison with Table A4. In summary, sampling noise accounts for 34% of single-call developer-level gain variance on AIME, developer-specific structure for 39%, and shared question signal for 27%, and the pooled seven-developer estimate has a two-run split-half correlation of 0.943 (Table A5); the corresponding noise shares are not measured on MMLU-Pro or GPQA-Diamond.

A.13 Budget-constrained allocation procedure

For each benchmark and each held-out developer, every question receives a score per policy:

- **Value:** mean paired gain across the six donor developers.
- **Difficulty:** donor non-thinking solve rate, ordered from lowest to highest.
- **Difficulty-fit:** gain predicted by the donor-fitted category-specific quartic model.
- **Category:** donor mean gain for the question's category.
- **Per-token variants:** value or difficulty-fit score divided by donor mean additional completion tokens, floored at one token.
- **Hindsight:** the held-out developer's realized gain, used as a descriptive reference.

Questions are sorted by score using deterministic tie-breaking (ties break by question identifier). Evaluation cost is the held-out developer's realized additional completion-token count, with negative counts floored at zero; the always-thinking token total uses the same zero floor. Completion-token counts are the completion-token field of the archived usage records. Every paired response was collected through a single gateway, so this field is reported uniformly across developers; it counts all generated tokens, including reasoning tokens where a model emits them (logged separately in the archive), and no response has a missing or zero count. Prompt tokens are not part of the cost. We accumulate costs along the ranking and linearly interpolate the resulting gain–cost curve on a common budget grid; gain fractions are cumulative held-out paired gains divided by the always-thinking total gain. Held-out costs and gains only score a completed allocation; every ranking uses donor information alone. Curves are averaged over the seven folds with equal weight. Random allocation is simulated by averaging 1,000 uniformly random orderings per fold under the same stopping and interpolation rules; the simulated curves recover 0.498 and 0.500 of the gain at the 50% budget, essentially the diagonal. The donor-based operating rule enables thinking exactly on the questions with positive predicted value (or positive difficulty-fit prediction) and is evaluated unchanged on the held-out developer. The fold-averaged value curves reach retrospective maxima of 106% and 103% of the always-thinking gain, at 84% and 92% of the additional tokens; these maxima are selected from the evaluation curves and are descriptive. Table A6 reports every policy's gain fractions at fixed budgets, Table A6b lists per-developer absolute gains, and Figure 5 shows the same fold-averaged policies as absolute accuracy against mean total completion tokens. At the 50% budget, the gain-fraction gaps between value and difficulty-fit rankings are 0.21 $[0.09, 0.34]$ on MMLU-Pro and 0.15 $[0.04, 0.27]$ on GPQA-Diamond, and 0.08 $[0.03, 0.16]$ and 0.05 $[-0.02, 0.12]$ for the per-token variants. In absolute accuracy, the per-token gap at the same budget is 0.011 $[0.005, 0.019]$ on MMLU-Pro and 0.010 $[-0.002, 0.021]$ on GPQA-Diamond. Negative additional-token counts, floored at zero, occur for 0.10% and 0.22% of held-out question cells. The 95% intervals for the value-minus-difficulty-fit gap and the per-token gap (value per token minus difficulty-fit per token) at the 50% budget, the positive-value rule's gain fraction, and the descriptive peak of the fold-averaged value curve use a 1,000-replicate question-cluster bootstrap that re-sorts each fold's fixed training-fitted scores in every replicate; the scores and difficulty surfaces themselves are not refit.

Table A6: Fraction of the always-thinking accuracy gain recovered at fixed fractions of the always-thinking extra completion tokens, averaged over LODO folds.

Benchmark	Policy	25%	50%	75%	Tokens for 90% gain
MMLU-Pro	Thinking value	0.397	0.797	1.014	0.58
MMLU-Pro	Value per token	0.634	0.918	1.032	0.48
MMLU-Pro	Difficulty-fit gain	0.309	0.588	0.868	0.78
MMLU-Pro	Difficulty-fit gain per token	0.607	0.835	0.978	0.62
MMLU-Pro	Difficulty (hardest first)	0.139	0.359	0.676	0.87
MMLU-Pro	Hindsight gain ranking	0.861	1.130	1.159	0.28
GPQA-Diamond	Thinking value	0.377	0.677	0.943	0.70
GPQA-Diamond	Value per token	0.529	0.791	0.953	0.64
GPQA-Diamond	Difficulty-fit gain	0.291	0.528	0.779	0.90
GPQA-Diamond	Difficulty-fit gain per token	0.466	0.744	0.907	0.74
GPQA-Diamond	Difficulty (hardest first)	0.137	0.392	0.634	0.93
GPQA-Diamond	Hindsight gain ranking	0.753	1.080	1.140	0.34

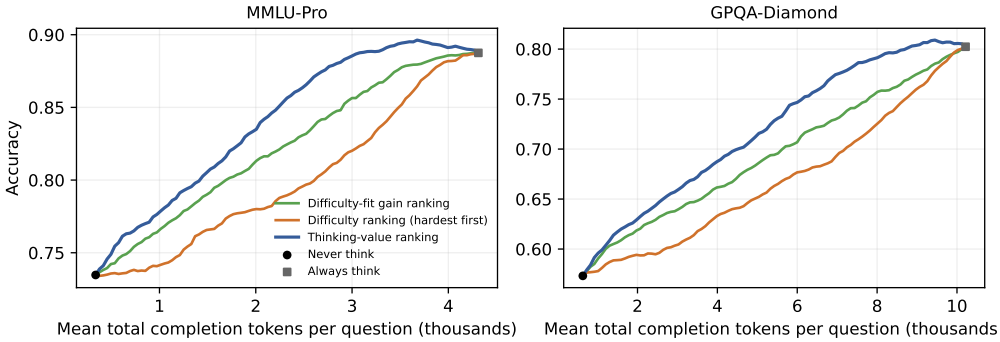

Figure 5: Absolute accuracy against mean total completion tokens per question (fold averages). Endpoints mark never-think and always-think; curves interpolate the allocation policies of Figure 4.

Table A6b: Per-developer absolute gains: always-thinking gain and the gain realized by value ranking at half the additional-token budget.

	MMLU-Pro		GPQA-Diamond	
Held-out developer	Always think	Value at 50%	Always think	Value at 50%
Alibaba	0.251	0.186	0.287	0.172
DeepSeek	0.143	0.113	0.239	0.138
Google	0.038	0.030	0.123	0.090
Mistral	0.314	0.198	0.355	0.202
Moonshot	0.054	0.054	0.123	0.084
OpenAI	0.096	0.074	0.134	0.106
Zhipu	0.172	0.150	0.343	0.270

A.14 Trace-length diagnostic

For each held-out developer, additional completion tokens (thinking-mode minus non-thinking-mode) are averaged over its model variants and prompts for each MMLU-Pro question. Difficulty and thinking value are estimated from the other six paired developers, not from the ten-model panel used elsewhere. Across folds, the median correlation between additional tokens and donor-estimated non-thinking difficulty is $r = 0.563$, against $r = 0.057$ with the held-out developer's realized gain, and -0.071 after a descriptive flexible adjustment. The controlled statistic residualizes the held-out developer's tokens and gains against a category-specific quartic in that donor-estimated difficulty with coefficients fit on the held-out observations themselves; it is a descriptive within-fold adjustment, not a training-only prediction. Realized token count may reflect solution progress, stopping, verbosity, or failure and is therefore not an assigned compute dose, and these associations do not identify the effect of assigning a larger reasoning budget.

A.15 VARIANCE DECOMPOSITION

The complete 18×300 matrix permits a balanced two-way decomposition. Sums of squares are computed for model main effects, question main effects, and the remaining interaction term: 9.7% of observed variance falls to model main effects, 30.4% to question main effects, and 59.9% to interaction plus measurement noise; on AIME, where repeats exist, roughly half of the analogous bucket is measurement noise, though the MMLU-Pro share is not directly measured. In the nested fixed-effect description, model explains 9.7%; category adds 4.5 percentage points; quadratic difficulty adds 10.5; and question identity a further 15.4. Both decompositions attribute variation to question identity beyond model-level mean differences; the remaining interaction term includes model-specific variation and sampling noise. MMLU-Pro has no repeated calls within each model–question–prompt–mode cell, so its interaction term also contains sampling noise. Appendix A.12 analyzes these components separately on AIME. At the developer level ($7 \times Q$, matching the primary estimand's weighting), question main effects account for 34.1% of variance on MMLU-Pro and 29.2% on GPQA-Diamond, developer main effects for 10.7% and 7.6%, and interaction plus noise for the remainder.

A.16 UNSEEN-QUESTION TEXT MODEL

Question folds are deterministically stratified within category. For each target developer and fold, training excludes both that developer and every question in the evaluation fold. Features include category; word, character, digit, equation-marker, and LaTeX-marker counts; a small set of structural indicators; and word/bigram TF-IDF features. A ridge model with fixed regularization predicts developer-level gain. Regularization is fixed before evaluation (ridge penalty $\alpha = 20$, five deterministic category-stratified question folds), and held-out-developer outcomes are not used for tuning. A companion baseline replaces TF-IDF with frozen 1,024-dimensional text embeddings (cached from OpenAI `text-embedding-3-large` before this analysis) and a ridge head under the same held-out-developer and held-out-question protocol; pooled correlations are 0.131 $[0.067, 0.188]$ on MMLU-Pro and -0.115 $[-0.179, -0.045]$ on GPQA-Diamond.

A.17 ESTIMATION LEDGER

For the reported statistics: (i) for LODO transfer and allocation, category means, difficulty controls, IRT-style parameters, and allocation scores are estimated from donor developers only, and split correlations fit controls within each developer group as specified in Section 3; (ii) difficulty controls use training-developer non-thinking solve rate or the external panel, never held-out outcomes; (iii) adjusted transfer correlations residualize both the predictor and the held-out outcome against the same training-fitted baseline; the trace-length diagnostic is the labeled exception, with a descriptive within-fold adjustment; (iv) bootstrap replicates resample question identifiers and retain each question's full vector of held-out outcomes; transfer and allocation intervals condition on the training-fitted baselines, scores, and residuals rather than refitting them per replicate, while the random-effects reference-correlation intervals refit their variance components; (v) permutation tests shuffle item-level residual predictions within held-out developer, category, and quintiles of the training-fitted baseline prediction, and are anti-conservative under the overlapping-donor design per the calibration in Appendix A.18; (vi) uncertainty is conditional on the seven-developer roster and three-prompt protocol, and the 35 split partitions overlap rather than replicate independently.

A.18 NULL SIMULATIONS

Ability-shift null. We fit two-parameter logistic item curves to the real non-thinking responses and give every model one ability and one thinking-induced ability shift, solved so the null reproduces that model's observed non-thinking and thinking accuracy exactly. No question carries a thinking-specific parameter. Binary outcomes are simulated under the real roster and call counts, with prompts as exchangeable replicates, and the complete transfer pipeline (developer-equal aggregation, splits, LODO, flexible and IRT-style adjustment) runs on each of 200 replicates per benchmark. The null's median (95th percentile) pooled LODO correlations are 0.365 (0.411) under flexible and 0.243 (0.292) under IRT-style adjustment on MMLU-Pro, and 0.460 (0.498) and 0.323 (0.365) on GPQA-Diamond; the observed values are 0.306, 0.264, 0.285, and 0.273. Split medians behave the same way, with only MMLU-Pro's IRT-adjusted split median (0.470) marginally above

the null's 99th percentile (0.469). The simulation treats prompts as exchangeable replicates and does not reproduce the full prompt variation in the observed data. It is therefore a diagnostic of the controls, not a complete account of the observed responses. Its comparable adjusted correlations show that these controls do not distinguish a separate thinking-specific item dimension from ability shifts along shared item response curves. Joint correctness models across modes and experiments with assigned budgets could help distinguish these explanations.

Permutation calibration. Under a null in which each developer's gains are its own difficulty-shaped mean curve plus independent noise, with no shared item-level signal, the pooled conditional permutation test rejects at rate 0.115 (MMLU-Pro) and 0.095 (GPQA-Diamond) at nominal 0.05, and 0.03 at nominal 0.01, over 200 replicates of 1,000 permutations. Independent shuffles within held-out developers break the dependence among the overlapping donor-mean predictors, making the test anti-conservative by roughly a factor of two at conventional levels. We therefore do not interpret the original permutation p-values as calibrated significance levels. We summarize uncertainty using the question-cluster bootstrap, which retains each question's full vector of held-out outcomes.

B Additional Transfer Results

Figure 6 repeats the raw-phenomenon strip on GPQA-Diamond, Figure 7 shows the variance decomposition and prompt agreement, Figure 8 shows LODO calibration and per-developer error reduction, Figure 9 plots thinking value against difficulty for every question, and Table A7 summarizes all 35 MMLU-Pro developer splits.

Within-developer temporal transfer. The roster contains two generations from one developer: five Qwen3 and five Qwen3.5 models. Question gains estimated from Qwen3 predict Qwen3.5 gains at raw $r = 0.660$ on MMLU-Pro and 0.414 on GPQA-Diamond (0.507 $[0.397, 0.601]$ and 0.360 $[0.216, 0.490]$ after the flexible control fit to the six non-Alibaba developers), although mean gains roughly double across the generation (0.176 to 0.326 and 0.194 to 0.380). The six unrelated developers predict Qwen3.5 comparably well (adjusted 0.435 and 0.462), and Qwen3's predicted top and bottom quartiles differ in Qwen3.5 gain by 0.645 and 0.273. The ranking transfers between the evaluated Qwen3 and Qwen3.5 model sets.

Donor-count sensitivity. With k donor developers supplying the item predictor (all k-subsets of the six training developers, with the same donor-fitted flexible baseline subtracted from both sides as in the primary LODO analysis), the median adjusted held-out correlation rises smoothly with k: 0.19, 0.25, 0.29, 0.32, 0.33, and 0.34 for $k = 1, \ldots, 6$ on MMLU-Pro, and 0.19, 0.23, 0.25, 0.27, 0.29, and 0.32 on GPQA-Diamond; at $k = 6$ the values reproduce the primary per-developer medians. With one donor, the median adjusted correlation is 0.19 on both benchmarks, compared with 0.34 and 0.32 with six donors. At the 50% additional-token budget, the median recovered-gain fraction is 0.65 on MMLU-Pro and 0.60 on GPQA-Diamond with one donor developer, increasing to 0.77 and 0.68 with six. Transfer can therefore support allocation with fewer donor developers, although evaluation cost also depends on their model variants and token usage.

Comparison with positive-flip prediction. We compare signed gains with a positive-flip predictor: the donor-developer mean rate of transitions from an incorrect non-thinking answer to a correct thinking answer. The flip predictor transfers to held-out developers, with median adjusted LODO correlation 0.27 on both benchmarks, compared with 0.34 and 0.32 for signed gains. At half the additional-token budget, it recovers 70% and 60% of the always-thinking gain, compared with 80% and 68% for signed value. These point estimates favor accounting for regressions as well as improvements.

Illustrative benchmark questions. Three archived MMLU-Pro items, selected by deterministic rules, illustrate the estimand; parentheses give the non-thinking solve rate, seven-developer thinking value, and developers with positive gain. Quotations preserve the benchmark text with typography lightly normalized.

- *Chemistry* (0.338; +0.633; 7 of 7): "A 0.100 l container maintained at constant temperature contains 5.0×10^{10} molecules of an ideal gas. How many molecules remain if the

volume is changed to 0.005 l? What volume is occupied by 10,000 molecules at the initial temperature and pressure?" Thinking increases recorded accuracy for all seven developers on this question.

- *Engineering* (0.476; −0.476; 0 of 7): "Find the steady-state solution of the equation $y[n] + [1/\sqrt{3}]\, y[n-1] = \cos(\pi/6)\, n$," followed by ten nearly identical trigonometric options. The mean thinking value is -0.476, and none of the seven developers has a positive gain.
- *Engineering* (0.067; +0.014; 1 of 7): "Determine the number of belts required to transmit 70 hp by a 3V large capacity belt of length 100 in., running on pulleys 12 in. in diameter and speed 2000 rpm." This question has both a low direct-answer solve rate and a small average gain from thinking.

Table A7: Summary of the 35 MMLU-Pro developer splits. Values summarize correlations across questions.

Statistic	Mean	Median	Minimum	Maximum
Pearson, raw	0.596	0.636	0.384	0.697
Spearman, raw	0.629	0.669	0.430	0.733
Pearson, within category	0.588	0.609	0.400	0.676
Pearson, category + difficulty adjusted	0.600	0.606	0.466	0.698

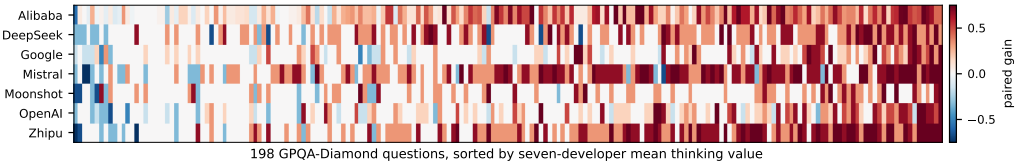

Figure 6: Raw-phenomenon strip on GPQA-Diamond: developer-level paired gains for all 198 questions, sorted by the seven-developer mean.

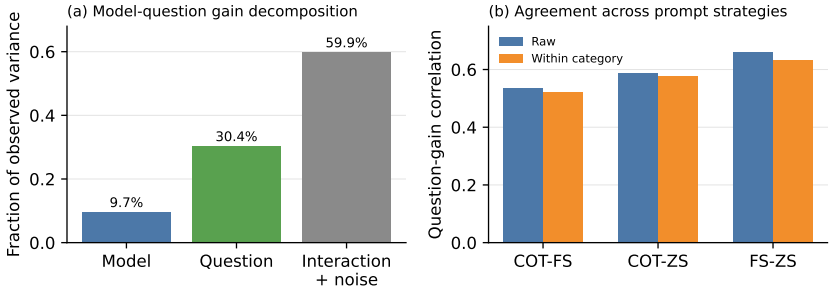

Figure 7: (a) Balanced decomposition of model-question gains. (b) Question-level gains agree across prompt strategies.

C Category-Level Results

Thinking value has interpretable coarse structure. Across the six MMLU-Pro categories, model-equal mean paired gain (18 models weighted equally, unlike the developer-equal primary estimand) ranges from $+0.061$ in biology to $+0.270$ in chemistry, with physics and mathematics near $+0.26$ and computer science and engineering near $+0.15$; each category has positive mean gains for at least 17 of the 18 models, and gains also vary across difficulty strata and GPQA-Diamond subdomains (Figures 10 and 11; Tables A8–A10). Restricting to the 265 ten-option items leaves the coarse structure unchanged (biology $+0.052$, chemistry $+0.287$). Question identity carries signal beyond category: independent groups still agree at $r = 0.606$ after flexible category-specific adjustment.

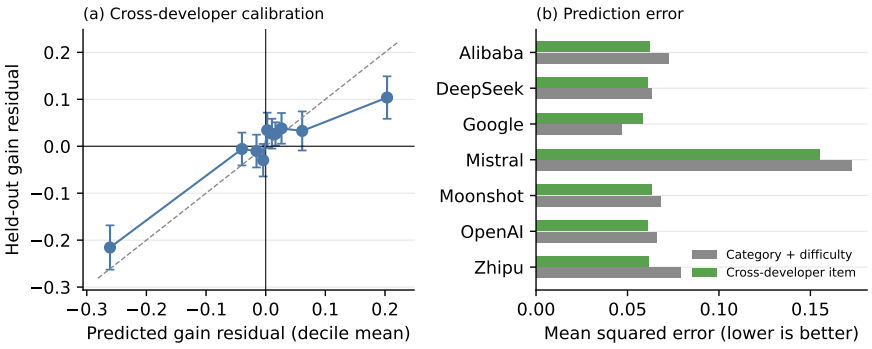

Figure 8: Additional LODO diagnostics on MMLU-Pro. (a) Predictions and outcomes after removing the training-fitted category/difficulty baseline, binned by predicted residual; error bars show 1.96 standard errors and are descriptive because outcomes share question IDs. (b) Cross-developer item prediction reduces MSE for six developers; Google is the exception.

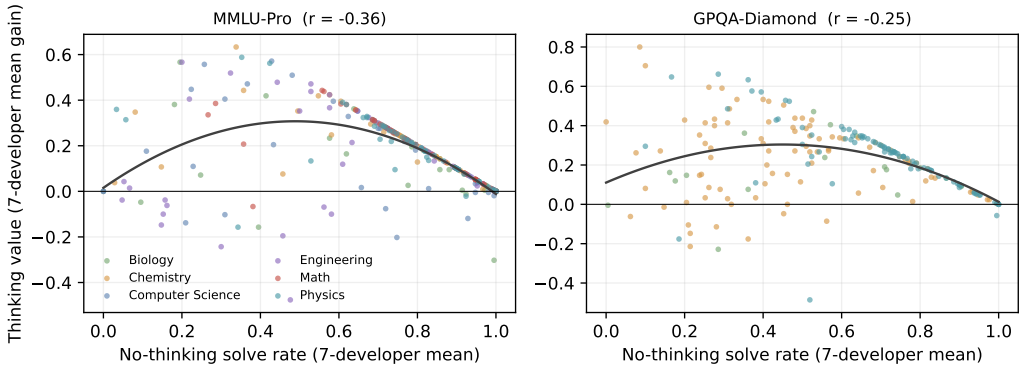

Figure 9: Thinking value against non-thinking solve rate for every question (7-developer equal-weight means). The fitted quadratic illustrates the overall difficulty trend; the transfer analyses instead use category-specific quartic controls. The transfer tests ask whether a component of the vertical spread at fixed solve rate reproduces across developers (the spread also contains noise and developer-specific variation). The upper-right edge reflects the constraint $\widehat{\tau}_q \leq 1-$ solve rate.

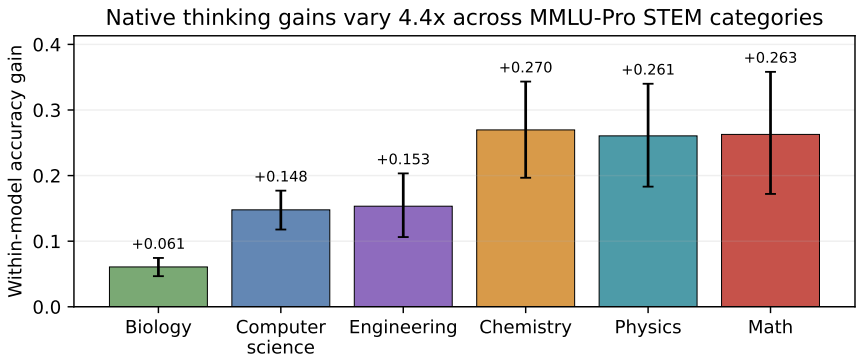

Figure 10: Mean paired thinking gain across 18 models on six MMLU-Pro STEM categories. Error bars are 95% model-bootstrap intervals.

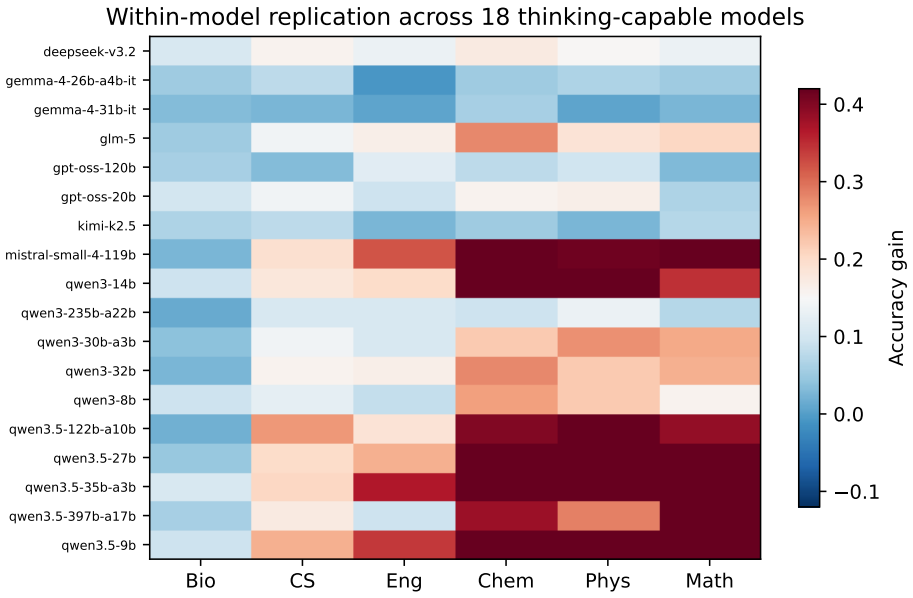

Figure 11: Model-by-category paired gains. Gains vary by model and category; each category has positive mean gains for at least 17 of the 18 models.

Table A8: MMLU-Pro category results, 18 models weighted equally. Model-bootstrap CIs resample models, so variant-rich families carry more weight than in the developer-equal analyses. "Positive" counts models with positive mean paired gain.

Category	No-think	Think	Gain [95% CI]	Positive
Biology	0.827	0.888	0.061 [0.047,0.074]	18/18
Computer science	0.676	0.824	0.148 [0.118,0.177]	18/18
Engineering	0.602	0.756	0.153 [0.106,0.203]	17/18
Chemistry	0.651	0.921	0.270 [0.197,0.343]	18/18
Physics	0.671	0.932	0.261 [0.183,0.340]	18/18
Mathematics	0.697	0.959	0.263 [0.172,0.358]	18/18

Table A9: MMLU-Pro gain by category and non-thinking difficulty tertile. Tertiles are global rank-based thirds of the 18-model mean non-thinking solve rate (100 questions each); cells are model-equal mean gains.

Category	Hard	Medium	Easy
Biology	0.183	0.167	0.002
Chemistry	0.376	0.280	0.039
Computer science	0.290	0.154	0.016
Engineering	0.197	0.177	0.050
Mathematics	0.387	0.308	0.050
Physics	0.406	0.253	0.031

Table A10: GPQA-Diamond paired gains by broad metadata subdomain.

Subdomain	Questions	No-think	Think	Gain [95% CI]
Biology	19	0.554	0.697	0.142 [0.100,0.187]
Chemistry	93	0.437	0.650	0.213 [0.160,0.266]
Physics	86	0.604	0.910	0.306 [0.236,0.374]

D Prompt Results

CoT prompting and native thinking show partial substitution in chemistry and physics, an exploratory comparison across six categories in which the chemistry interval barely excludes zero.

The native gain under zero-shot prompting exceeds the native gain under CoT prompting by 0.044 in chemistry (model-bootstrap 95% CI $[0.001, 0.093]$) and 0.064 in physics ($[0.021, 0.123]$); other categories have intervals spanning zero (Table A12). The smaller incremental gain from native thinking under CoT prompting is consistent with partial overlap between the two strategies in these categories.

Table A11: Question-level agreement across prompt strategies (developer-equal question gains per prompt; correlations across questions).

Prompt pair	Pearson	Within-category Pearson	Spearman
CoT–few-shot	0.535	0.523	0.592
CoT–zero-shot	0.587	0.577	0.633
Few-shot–zero-shot	0.660	0.632	0.696

Table A12: Reduction in native thinking gain under CoT versus zero-shot prompting. Positive values indicate partial substitution.

Category	Reduction	95% model-bootstrap CI
Biology	−0.010	[−0.033,0.014]
Chemistry	0.044	[0.001,0.093]
Computer science	0.025	[−0.009,0.061]
Engineering	0.026	[−0.014,0.069]
Mathematics	0.036	[−0.008,0.094]
Physics	0.064	[0.021,0.123]

E Category Allocation Details

The category policy enumerates every subset of the six categories. In each leave-one-model-out fold it estimates category gain and additional-token cost from the other 17 models, then chooses the feasible subset with maximum estimated gain. At a 50% target it always selects chemistry, mathematics, and physics, realizing 48.7% of the token overhead and 64.4% of always-thinking gain. At 25%, it realizes 23.6% of tokens and 32.4% of gain; at 75%, 71.5% of tokens and 87.4% of gain. The policy uses public category labels and is evaluated on the same question distribution. Its leave-one-model-out results are descriptive and are not directly comparable to the leave-one-developer-out curves in Section 6.

F Exploratory Open-Format Analysis: AIME

The archive contains a complete paired design on all 90 AIME 2024–2026 questions: the same 18 models, seven developers, and three prompts, with five runs per cell averaged before contrasts. AIME is open-ended (integer answers) rather than multiple choice, and it is the regime where provider compute envelopes bind: 3.07% of thinking responses are truncated (versus 1.60% non-thinking), and 69 of 90 questions have at least one truncated response somewhere in the repeated matrix. Invalid-extraction rates are higher in non-thinking mode: 0.83%, compared with 0.05% in thinking mode. The sensitivity rows in Table A13 filter at the response level: flagged responses are removed and questions retaining at least one valid call per model–prompt–mode cell are kept, which is why 69 truncation-touched questions leave 78 evaluable items.

Table A13: AIME transfer and sensitivities (developer-equal question gains; CIs cluster by question). The mean developer-equal thinking gain is 0.374.

Analysis	Items	Median split r	Pooled LODO r [95% CI]
Raw agreement	90	0.732	0.605 (raw)
Flexible year-difficulty control	90	0.317	0.157 [0.059,0.239]
IRT-style control	90	0.327	0.158 [0.058,0.236]
External panel difficulty (rel. 0.958)	90	0.633	0.290 [0.153,0.392]
Developer-disjoint panel (rel. 0.922)	90	0.648	0.324 [0.207,0.416]
Remove invalid-extraction responses	87	0.331	0.155 [0.055,0.235]
Remove truncated responses	78	0.183	−0.028 [−0.077,0.011]
Request-matched subset	90	0.029	0.063 [−0.022,0.145]
Request-matched + external difficulty	90	0.584	0.287 [0.166,0.391]

AIME transfer is positive under the outcome-independent difficulty panels: gains are the largest of any benchmark, the raw developer-split agreement (0.732) is the highest, and the held-out quartile contrast is 0.414. The transfer estimate is sensitive to the analysis: after removing truncated responses, the pooled adjusted correlation is -0.028 (95% CI $[-0.077, 0.011]$); the internally adjusted request-matched estimate is also weak, and estimates vary across years (pooled LODO r of 0.218 $[-0.008, 0.347]$, 0.164 $[0.052, 0.254]$, and 0.051 $[-0.068, 0.154]$ for 2024, 2025, and 2026, on 30 items each). Truncation is an operational outcome of the thinking policy and touches most items (69 of 90 questions), so shared compute-envelope behavior plausibly contributes to the cross-developer agreement here; because truncation filtering conditions on a post-treatment outcome and changes the evaluated items, the reanalysis cannot say how much. AIME provides a sensitivity analysis in an open-ended, truncation-prone setting. Experiments with assigned reasoning budgets are needed to evaluate transfer under controlled token limits. Full per-fold and per-sensitivity outputs accompany the submission.

G Scope of Claims

The primary results concern the configured-mode accuracy contrast on 300 MMLU-Pro STEM and 198 GPQA-Diamond questions under three prompts. Transfer persists across held-out developers after adjustment for the fitted correctness measures; an ability-shift response-curve null reproduces adjusted transfer of comparable magnitude, so the results establish transferable, allocation-relevant structure rather than a dimension beyond response-curve accounts. On these evaluated questions, value ranking exceeds the matched difficulty-fit allocator; after cost normalization, the improvement is statistically resolved on MMLU-Pro but not on GPQA-Diamond. Inference is conditional on the evaluated roster, protocol, and fitted controls. The study does not establish prediction for unseen questions, isolate the effect of reasoning-token budgets, or evaluate end-to-end deployment efficiency.

H Intervention and Request Audit

Table A14 reports every paired configuration used in the benchmark analyses. Rows combine model sizes only when the stored native-mode switch, sampling fields, and system-message behavior are identical. All requests use OpenRouter with provider fallback disabled, and the provider-returned model identifier is mode-invariant for every model. "Default" means the field was omitted and therefore left to the provider; no explicit maximum-token field or random seed was supplied. Direct mode includes the fixed system message "You are a helpful assistant," whereas native-thinking mode omits the system role according to the archived configurations. The seven-model sensitivity matches the logged sampling fields across modes. System messages still differ, and omitted fields may be resolved differently by the provider.

Table A14: Audit of paired native-mode configurations. Sampling is $(\text{temperature}, \text{top-}p, \text{top-}k)$, shown as direct $\rightarrow$ thinking. All returned identifiers were stable across modes.

Model(s)	Native switch	Sampling, direct $\rightarrow$ think	Returned identifier audit
DeepSeek-V3.2	`reasoning`: off $\rightarrow$ on	$(1, .95, -) \rightarrow (1, .95, -)$	dated suffix 20251201
GLM-5	`reasoning`: off $\rightarrow$ on	default $\rightarrow$ default	dated suffix 20260211
Kimi-K2.5	`reasoning`: off $\rightarrow$ on	$(.6, .95, -) \rightarrow (1, .95, -)$	dated suffix 0127
Mistral-Small-4-119B	effort: none $\rightarrow$ high	$(.7, \text{def.}, \text{def.})$ in both	`mistral-small-2603`
Gemma-4-26B-A4B, 31B	`reasoning`: off $\rightarrow$ on	$(1, .95, 64)$ in both	suffixes 20260403 / 20260402
GPT-OSS-20B, 120B	effort: low $\rightarrow$ high	default $\rightarrow$ default	requested IDs returned unchanged
Qwen3: 8B, 14B, 30B-A3B, 32B, 235B-A22B	`reasoning`: off $\rightarrow$ on	$(.7, .8, 20) \rightarrow (.6, .95, 20)$	dated suffix 04-28
Qwen3.5: 9B, 27B, 35B-A3B, 122B-A10B	`reasoning`: off $\rightarrow$ on	$(.7, .8, 20) \rightarrow (1, .95, 20)$	dated suffixes 20260310 / 20260224
Qwen3.5-397B-A17B	`reasoning`: off $\rightarrow$ on	$(.7, .8, 20) \rightarrow (.6, .95, 20)$	dated suffix 20260216