HINTED SCALING LAWS

Anonymous authors
Paper under double-blind review

ABSTRACT

Downstream scaling laws let developers predict benchmark performance at large scale from small, cheap proxies. This prediction fails when the small proxies score zero because their accuracies give no signal to extrapolate to larger scale. We introduce hinted scaling laws, which use privileged information to turn a benchmark into a continuous sequence of easier versions so that even the small proxies score above zero. We find that accuracy rises smoothly and predictably with the hint level, enabling new types of scaling laws that operate below the noise threshold. Within pretrained and post-trained model families, hinted scaling laws fit with the smallest models reduce the median forecasting error for larger models from 21 percentage points for a standard scaling law to 6. Hinting also improves the sample efficiency of model ranking. It widens the gaps in accuracy between models while approximately preserving their order, so a single hinted rollout per problem ranks near-indistinguishable models as accurately as 5 unhinted rollouts. Our results suggest that hinting is a general primitive for improving the accuracy and cost of downstream scaling laws and other prediction tasks in model development.

1 INTRODUCTION

By extrapolating the effect of a design choice from small, cheap runs to an expensive target scale, scaling laws (Kaplan et al., 2020; Hoffmann et al., 2022) enable affordable tuning of data and algorithmic choices for large-scale performance. This has worked best in pretraining because loss varies predictably with scale, so even very small models carry information about larger ones.

Yet pretraining loss is an imperfect proxy for the downstream capabilities we ultimately care about, because lower loss does not reliably mean better task performance (Liu et al., 2023; Springer et al., 2025; Watts et al., 2026). Thus, we want scaling laws that directly predict downstream performance. Such laws would let developers optimize design decisions for downstream tasks at a fraction of the target scale, and help safety teams anticipate when a capability will emerge while current models still fail at it. Unfortunately, benchmark performance does not vary predictably with scale like pretraining loss does. Small models score at chance on challenging benchmarks, which provides no signal for extrapolation because a sequence of zeros cannot identify a scaling law. As a result, scaling laws fit directly to accuracy are often unreliable (Lourie et al., 2025).

We find that benchmark accuracy does vary predictably with a second variable: how much *help* the model is given. We formalize this help as *Benchmark Hinting*. Given a benchmark, Benchmark Hinting constructs a continuous sequence of easier benchmarks indexed by a hint level H, with the original benchmark at $H = 0$. We instantiate three hinting methods which each make the benchmark easier using privileged information such as a worked solution to each problem (Fig. 1, top).

Benchmark Hinting produces two useful phenomena. First, accuracy rises smoothly with H even for models that score below the noise threshold on the original benchmark, which enables evaluation on benchmarks where every model scores zero. Second, accuracy has a predictable joint structure over *both* hint level H and model scale, with increasing H acting similarly to increasing model scale (§3).

Given this shared structure between model scale and hint level, we are able to fit an accurate downstream performance scaling law even when the scaling ladder models score zero on the original benchmark (Fig. 1, bottom). A standard scaling law fits benchmark performance E as a function of model scale C alone. We instead fit a joint law $L(C, H)$ over scale and hint level. This approach of bootstrapping a sequence of easier benchmarks ($H > 0$) from a challenging target benchmark ($H = 0$) enables the joint law $L(C, H)$ to extract useful information from smaller model scales that

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

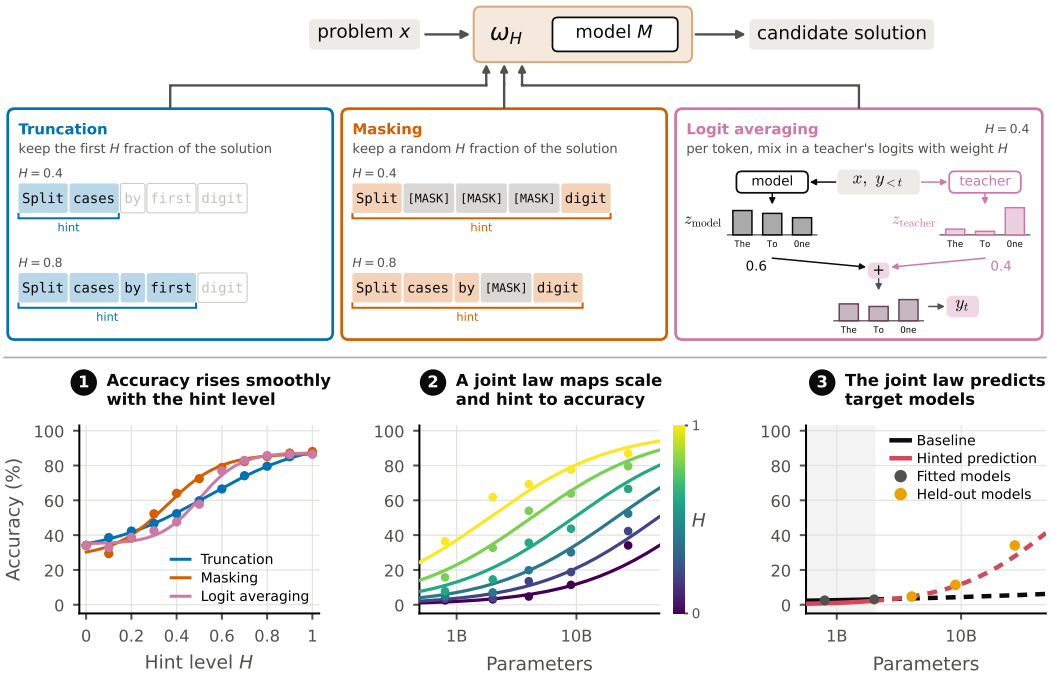

Figure 1: **Hinted scaling laws. Top:** Benchmark Hinting turns a benchmark into a continuous sequence of easier versions indexed by a hint level H. Hinting methods convert an answer sampling protocol ω into a hinted protocol ω_H. We instantiate three hinting methods: truncation, masking, and logit averaging. **Bottom:** results for the Qwen3.5 family on HLE. **(1)** Accuracy of Qwen3.5-27B rises smoothly with H, where $H = 0$ is the original benchmark. **(2)** A joint law in model scale and hint level (Eq. (1)) fits accuracy across the family. **(3)** A joint law fit to only the **two smallest models**, which score near zero at $H = 0$, gives a **hinted prediction** when evaluated at $H = 0$, forecasting the **held-out larger models**. A **baseline scaling law** fit to unhinted accuracy alone stays near zero.

achieve guessing performance on the original benchmark. We evaluate this joint law at $H = 0$ to map model scale to accuracy on the original benchmark and hence predict performance for runs at large C. Altogether, our contributions are as follows:

- **Methodology.** We introduce Benchmark Hinting, which turns a benchmark into a continuous sequence of progressively easier versions indexed by a hint level, and instantiate it with three hinting methods: truncation, masking, and logit averaging (§2).
- **Phenomena.** We show that under each hinting method, benchmark accuracy varies smoothly and predictably with the hint level, including for models that score zero on the original benchmark. We also find that hint level H and model scale form a simple joint structure, with increasing hint level H acting similarly to increasing model scale (§3).
- **Applications.** We use this structure in two applications. First, we fit a three-parameter joint law over scale and H that uses only the smallest models of pretrained and post-trained families. Forecasting the largest model's accuracy on the original benchmark, it reaches a median absolute error of 6%, against 21% for a standard scaling law, and is more accurate in 17 of 19 settings (§4). Second, because hinting widens the accuracy gaps between models while largely preserving their order, it ranks nearly indistinguishable models more accurately at small evaluation budgets. With one truncation-hinted rollout per problem, our ranking improves Kendall's τ over an unhinted ranking by 0.33 on AIME and 0.26 on HLE (§5).

2 Benchmark Hinting

We define a benchmark $B = (\mathcal{Q}, \omega)$ as a set of problems $\mathcal{Q}$, each with a way of grading a solution, and a protocol ω that takes a model M and a problem $x \in \mathcal{Q}$ and samples a candidate solution.

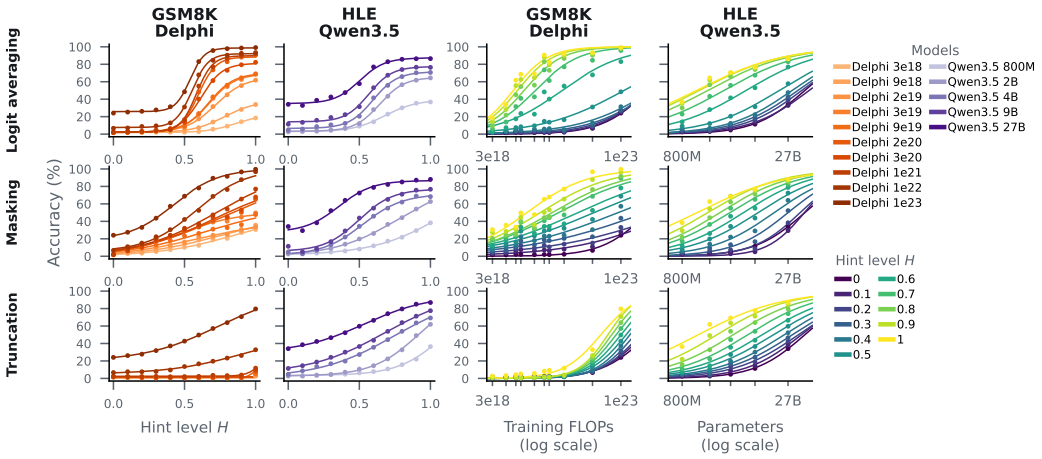

Figure 2: **Hinted benchmark accuracy varies smoothly in both hint level and scale.** Rows are the three Benchmark Hinting methods. Columns are the Delphi pretraining ladder (Held, 2026) on GSM8K and the Qwen3.5 family on HLE. **Left:** accuracy against hint level H, one curve per model, darker is larger. **Right:** accuracy against model scale C, one curve per hint level. Each curve is a sigmoid fit to a single model scale or hint level. As a function of H, the asymptotes are free: $L + (U - L)\,\sigma(a_1 H + a_0)$, with $0 \le L \le U \le 1$. As a function of C, they are pinned to 0 and 1: $\sigma(a_1 C + a_0)$, since a model of zero scale should score zero and an infinitely scaled one 100%.

ω fixes the prompt format and sampling method. We write $\mathrm{Acc}(M, B)$ for model M's expected accuracy on B over the randomness in ω. A *Benchmark Hinting method* maps B and a *hint level* $H \in [0, \infty)$ to a new benchmark $B_H = (\mathcal{Q}, \omega_H)$ with the same problems but a protocol that makes problems easier as H grows, for example by by letting a second model that has seen the solution influence token sampling. We abbreviate $\mathrm{Acc}(M, H) := \mathrm{Acc}(M, B_H)$.

We outline three basic desiderata for a hinting method. First, the benchmark at hint level $H = 0$ must match the original benchmark, so $\omega_0 = \omega$ and $B_0 = B$. Second, for a model M, $\mathrm{Acc}(M, H)$ should predictably increase in H according to some simple law. Third, for a fixed hint level H, $\mathrm{Acc}(M, H)$ should increase predictably with the capability of the model M, measured by a scale variable C such as pretraining FLOPs. We refer to the latter two desiderata as *hint scaling* and *capability scaling*. In §2.1, we present three instantiations of Benchmark Hinting. In §3, we provide evidence that each demonstrates hint and capability scaling. Then in §4, we show that these methods can be used to construct a standard downstream scaling law that enables us to predict large-scale model performance.

2.1 Instantiations of Benchmark Hinting

Truncation. In truncation, the protocol ω_H places the first H fraction of the reference solution, cut at a word boundary, in the model's context before it answers. We re-word the reference solution to not reveal the specific answer string (e.g. the numerical answer to an AIME problem).

Masking. In masking, ω_H places the reference solution in-context but replaces a randomly chosen $1 - H$ fraction of its words with `[MASK]` so that at $H = 1$ the solution is intact and at $H = 0$ every word is hidden. At $H = 0$ we drop the hint entirely since it would consist only of `[MASK]` tokens.

Logit Averaging. In logit averaging the model is decoded jointly with a *teacher*, a model that is given the reference solution in its context, and the next token is drawn from a weighted average of the two models' logits. Concretely, the two models share a vocabulary, and at each decoding step we write $z_{\mathrm{model}}(x, y_{<t})$ and $z_{\mathrm{teacher}}(x, y_{<t})$ for the logits that the evaluated model and the teacher, respectively, produce for the t-th answer position. We then form the mixed logits $s_H(x, y_{<t})_j = (1 - H)\, z_{\mathrm{model}}(x, y_{<t})_j + H\, z_{\mathrm{teacher}}(x, y_{<t})_j$ and sample $y_t \sim \mathrm{softmax}(s_H / T)$ at the evaluated model's temperature T. The sampled token is appended to both models' contexts and decoding continues. At $H = 0$, we simply sample from the z_{model}. In our experiments, the teacher is a second copy of the evaluated model whose context contains the reference solution (Appendix §C for details).

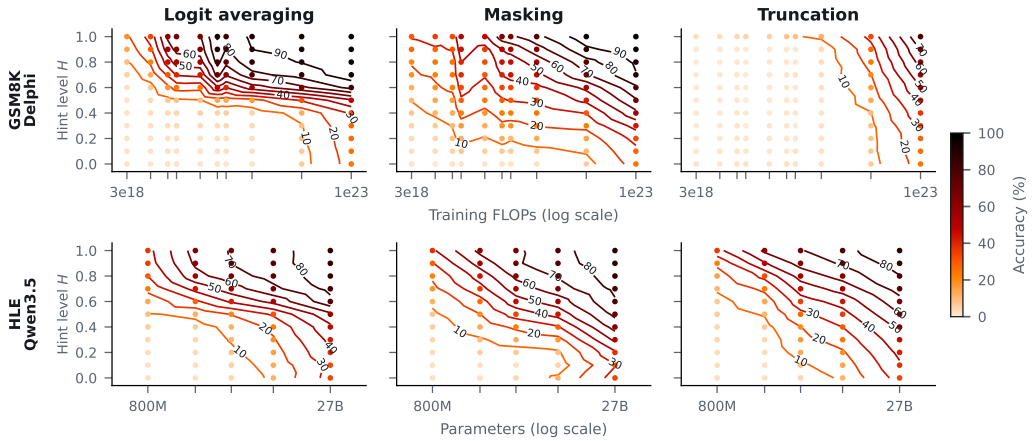

Figure 3: **Iso-accuracy contours in the scale and hint level plane.** Rows are the Delphi ladder on GSM8K and the Qwen3.5 family on HLE, columns the three Benchmark Hinting methods. Each dot is one measured cell, colored by accuracy. Lines are contours of constant accuracy at 10% to 90%, interpolated linearly between the cells with no law fitted. The slope of a contour is the exchange rate between scale and hint level at that accuracy.

3 Hinting and Capability Scaling

We study two core phenomena of Benchmark Hinting. First, accuracy increases smoothly as a function of the hint level H. Second, benchmark accuracy has a simple joint structure over both the hint level and the model scale where increasing hint level acts similarly to increasing model scale. We begin by measuring how benchmark accuracy varies with hint level and with model scale under the three Benchmark Hinting methods of §2.1. To do so we need model families, collections of models trained in the same way that differ only in a scale variable C. We consider pretraining scaling ladders (where C is training FLOPs) and released post-trained families (where C is model parameters).

Methods and data. For the pretraining ladder we use Delphi, an open-weight scaling ladder of pretrained language models which contains ten models spanning 447M to 25B parameters (Held, 2026). For post-trained ladders, we use four families: six dense Qwen2.5 Instruct models, six dense Qwen3 models, five dense Qwen3.5 models and five Gemma-3 instruction tuned models. For brevity, we display figures in this section for the pretrained Delphi ladder and the newest post-trained family, Qwen3.5 (see §D for the rest).

We evaluate the pretrained models on GSM8K and the post-trained models on AIME 2025 and 2026 (AIME) and a subset of Humanity's Last Exam (HLE), described in §A. On HLE we report only the Qwen3 and Qwen3.5 families because every member of the other two post-trained families scores around 0.

We evaluate all three Benchmark Hinting methods. Each requires a correct reference solution for every problem. For AIME and HLE we generate these solutions as described in §B, for GSM8K we use the answer strings that come with the benchmark. Every method sweeps eleven hint levels $H \in \{0.0, 0.1, \ldots, 1.0\}$, and at each H we draw 10 rollouts per problem to estimate the expected accuracy over rollout stochasticity.

Accuracy scales smoothly with hint level. In Fig. 2 (left) we show the accuracies of Delphi on GSM8K and Qwen3.5 on HLE at various hint levels. For every model, accuracy increases remarkably smoothly in H, and a sigmoid with free asymptotes fitted to each model fits very well. Larger models generally stay above smaller ones at every hint level, so *the family's ordering is preserved as H increases*. The shape of the rise is a property of the Benchmark Hinting method. Logit averaging helps sharply near $H = 0.5$ for every rung, masking climbs steadily across the whole range, and truncation moves little until most of the solution is shown.

Accuracy of hinted benchmarks scales smoothly with compute. The right half of Fig. 2 fixes the hint level, and we see at every level (1) the data follows a sigmoid very closely, and (2) a higher hint

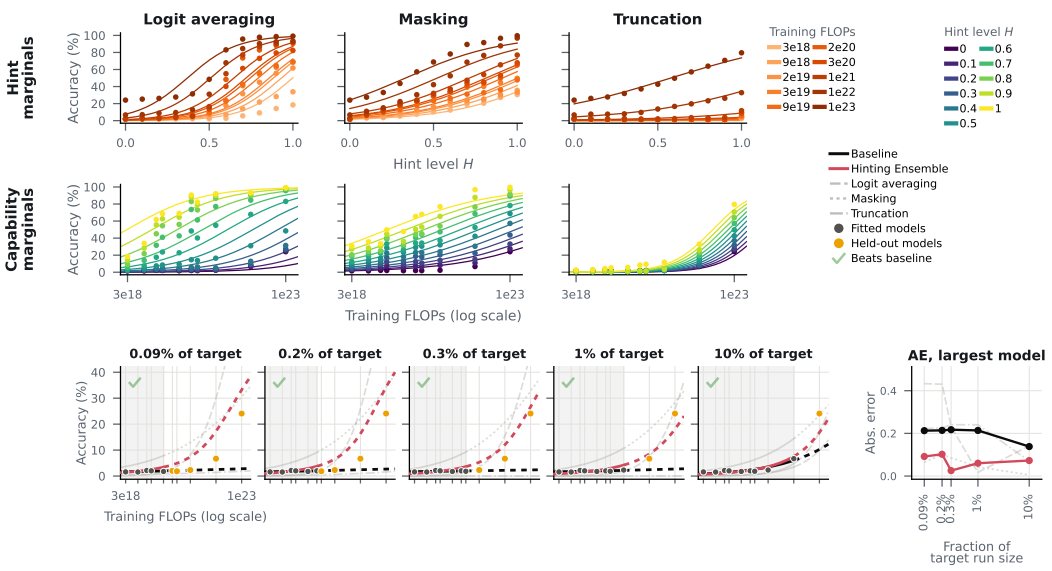

Figure 4: **Benchmark Hinting on the Delphi ladder (GSM8K). Top:** for each hinting method we fit one additive joint law (Eq. (1)) over all models and hint levels, with $\log_{10}$ pretraining FLOPs C. **Bottom:** the same law is refitted on only the t smallest models and its unhinted (H=0) slice is evaluated at the larger, held-out models. The **black curve** is the baseline which fits a sigmoid to unhinted accuracies alone. Each individual method's forecast is shown in **faint grey**, with the **red curve** being the average prediction over the methods. A ✓ marks panels in which the average forecast beats the baseline forecast on the largest ladder model; the right-most panel reports the absolute error (AE) against the size of the largest training model.

level shifts the curve left with curves rarely crossing. Increasing H thus moves the emergence point towards smaller models. For example, at $H = 0$ the Delphi ladder scores zero until 10^{22} FLOPs, while at $H \geq 0.5$ even the smallest model scores above zero for logit averaging and masking.

Hint level and capability exchange rate. We now move on to analyzing how accuracy varies as compute and hint level change *jointly*. The two marginals in Figure 2 are slices of one surface, namely accuracy as a function of scale and hint level. Figure 3 shows that surface directly. The lines indicate iso-accuracy contours, along which accuracy is constant. Reading along a contour gives an exchange rate between hint level and model scale. The slope of the contour gives the amount of scale one unit of H substitutes for at the contours accuracy. For masking and for every method on Qwen3.5 the contours run as roughly parallel diagonals. This constitutes a *very simple* joint structure in which hint level trades linearly for model scale at all accuracies. For logit averaging on Delphi the structure seems to be more complex. The contour slope changes with compute, with hints buying a lot at middle scales, but little at large and small scales.

4 Predicting Benchmark Performance

Having established that accuracy varies predictably with hint level and model scale, we turn to predicting the performance of *scaled up models*. We adopt the model families and benchmarks of §3. Our goal is to predict the accuracy of a family's largest members on the *unhinted* benchmark using measurements on smaller members. Concretely, we fit a single law $L(C, H)$ to all hint levels of the t smallest members of a family and evaluate its $H = 0$ slice on the held out larger members. When t is small this involves fitting few samples, putting us in a low signal to noise regime. Thus we prioritize a simple function form for L to prevent overfitting. To avoid cherry-picking a single method and to denoise our estimates, we use a simple *hinting ensemble*: we fit a separate law $L_\omega(C, H)$ for each of the three hinted protocols ω_H and predict with their mean, $L(C, 0) = \frac{1}{3}\sum_\omega L_\omega(C, 0)$.

Scaling law. To fit a scaling law $L(C, H)$ we must select a functional form for L. We can think of a Benchmark Hinting method and a functional form as a good pairing if two conditions hold. The

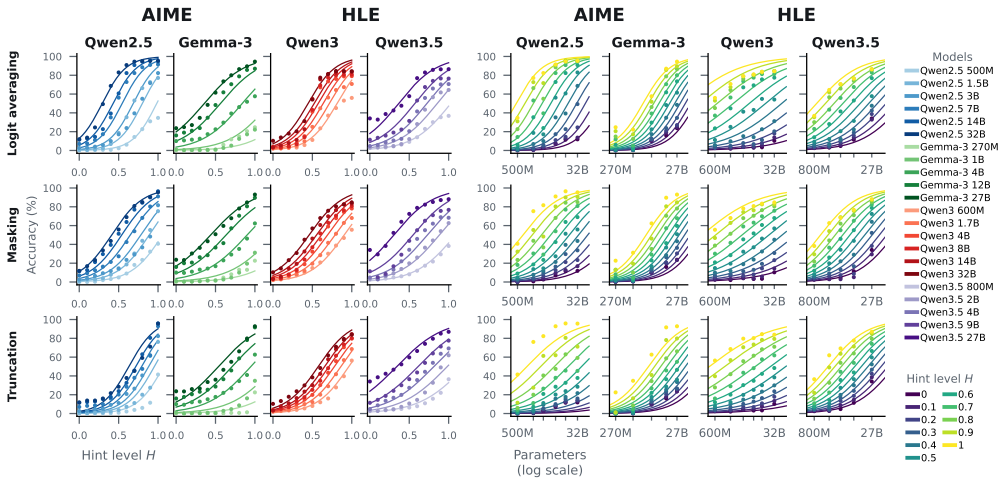

Figure 5: **Benchmark Hinting leads to scaling laws with low fit error.** Benchmark Hinting laws fit to post-trained model families. Rows correspond to Benchmark Hinting method, columns to model family and benchmark. For each (model family, Benchmark Hinting method) we fit a *single joint law* following Eq. (1) to the accuracy values of each model in the family on the original benchmark and hinted versions with $h \in \{0.1, 0.2, \ldots, 1\}$. **Left:** The hint marginals (fixing model scale and letting hint level vary) of each joint law. **Right:** The capability marginals (fixing hint level and letting model scale vary) of each joint law. Each joint fit is thus visualized as two plots, one on the left and right.

first is that L has *low fit error*. A single law $L(C, H)$ must fit data across all scales The second is that L has *low extrapolation error* when fit on small-scale model data. Fitting on small-scale model data with $H \neq 0$ included must pin down L well enough to predict the upscaled target.

From §3 we saw that accuracy was well modeled as a sigmoid of hint level and model scale. This suggests that $L(C, H)$ could follow a sigmoid of form:

$$L(C, H) = \sigma(\beta_1 C + \beta_2 H + \beta_3)\,, \qquad \sigma(z) = \frac{1}{1 + e^{-z}}, \tag{1}$$

where C is either $\log_{10}$ FLOPs or $\log_{10}$ parameter count, and $\beta_1, \beta_2, \beta_3$ are the fitted parameters. We note that this law enforces that hint marginals have 0 and 1 asymptotes, which is different to how we fit them in Fig. 2, and thus we believe this law to be an approximation of the true underlying relationship. Additionally, this law suggests that iso-accuracy curves of Fig. 3 have the same slope, $-\beta_1/\beta_2$, everywhere. From Fig. 3 we can see this is more reasonable for some (model, hint) method pairs than others. Most importantly, we find that in practice this law fits the data well (Table 1).

Methods and data. We use the model families, benchmarks and hinting methods of §3, with the same hint levels and rollouts per problem. Not every family makes for an informative forecasting problem on every benchmark, so we show a subset in the main body for brevity (all pairs in §E). In particular we exclude pairs if (1) every model in the ladder scores 0 on the $H = 0$ benchmark (making prediction trivial), or (2) if every model in ladder scores above 0 on the $H = 0$ benchmark. From this, we report Qwen2.5 and Gemma-3 on AIME, Qwen3 and Qwen3.5 on HLE, and Delphi on GSM8K.

Table 1: Mean absolute error of the fitted laws, averaged over models. The joint law uses three parameters per family where the independent marginals use $4|M| + 22$.

Method	Joint Fit	Individual Fit
Logit averaging	0.052	0.019
Masking	0.034	0.019
Truncation	0.030	0.015

4.1 Benchmark Hinting leads to scaling laws with low fit error

In Figs. 4 and 5 we demonstrate that the joint law in Eq. (1) has low fit error. For each benchmark, Benchmark Hinting method, and model family, we fit Eq. (1) to every model's accuracy at eleven

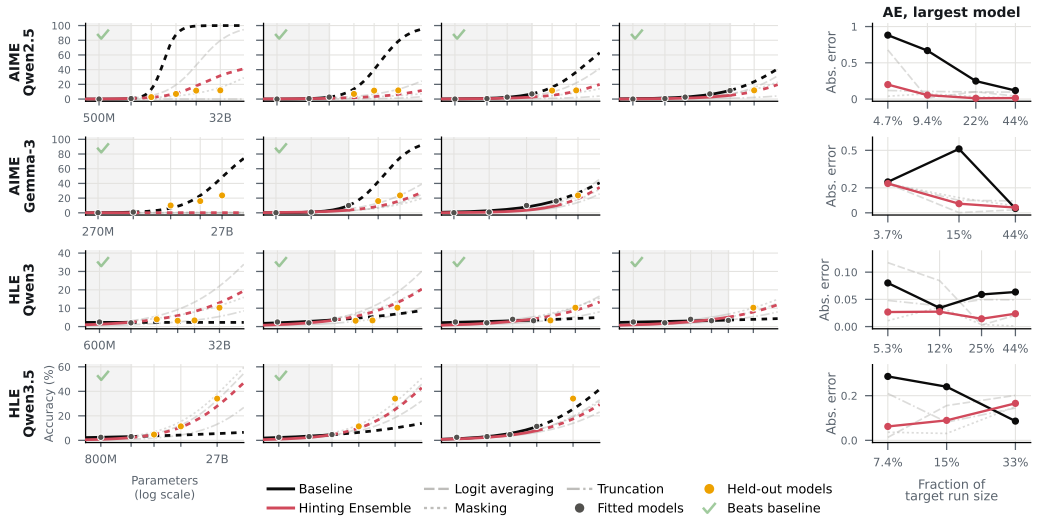

Figure 6: **Benchmark Hinting leads to scaling laws with low extrapolation error.** Benchmark Hinting forecasts of benchmark accuracy on post-trained model families. Rows correspond to model family and benchmark, columns to number of models used in the train set. For each Benchmark Hinting method we fit the joint law Eq. (1) to all hint levels of the train set models, and use the $H = 0$ marginal for the extrapolated curve. Each hint method is shown in **faint grey**, with the **red curve** averaging over methods. The **black curve** is the baseline sigmoid fit to original accuracies. A ✓marks panels in which our average forecast beats the baseline on the largest model in the family. The right-most column reports absolute error against the size of the largest train set model.

hint levels, $H \in \{0, 0.1, \ldots, 1\}$. We then draw the two marginals of that single fitted surface. Fig. 5 shows the results for the post-trained model families, and Fig. 4 top for the pretrained models.

Fitting every marginal on its own as we did in Fig. 2 requires $4|M| + 2|H|$ parameters, where $|M|$ is the number of models in the family and $|H|$ is the number of hint levels. This is between 42 and 62 for our families, in comparison to the three parameters for each joint fit. We compare the MAE (averaging across all families) of the joint fit against the individual marginal fit in Table 1 and find that the joint fit, despite using between 39–59 fewer parameters, increases MAE by at most 3.3 percentage points. §E gives a second view of the same point, drawing the iso-accuracy contours the fitted joint law implies next to the empirical ones of Fig. 3. The implied contours track the empirical ones. A more complex joint law is possible but the three parameter law already has low fit error so we keep it.

4.2 Benchmark Hinting leads to scaling laws with low extrapolation error

In Figs. 4 and 6 we demonstrate that the joint law has low extrapolation error when fit on a small number of models. We fit the joint law of Eq. (1) to all eleven hint levels of the t smallest members of a family, and evaluate its $H = 0$ marginal on the larger members. The baseline is a simple two-parameter sigmoid fit to the original benchmark accuracy on the same t smallest models. Both methods therefore see the same models and the same unhinted scores, and differ only in whether the hinted scores are available. Fig. 6 shows post-trained model results, and Fig. 4 pretrained.

In many cases the baseline fails because it is not identifiable from the fitted models. When the t smallest members of a family sit near zero on the unhinted benchmark, the fitted sigmoid is close to unconstrained leading to very poor extrapolation. On Qwen2.5 with two fitted models it reaches a held-out MAE of 0.644, predicting near-perfect accuracy at 32B where the true value is 13%. Our average forecast on the same two models reaches 0.069. Across all nineteen panels our average forecast beats the baseline in seventeen. We also compare to Pass@k as a hinting method (§E.2), and a trivial hinting baseline that randomly reveals the correct answer with probability H (§F), outperforming both. Overall we see that hinted measurements allow us to identify the benchmark accuracy scaling law in many cases where unhinted measurements alone do not.

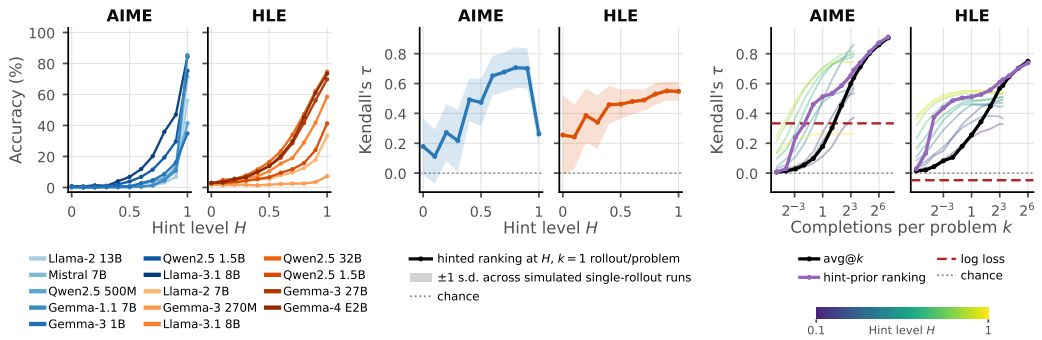

Figure 7: **Benchmark Hinting largely preserves the model ranking and recovers it from fewer rollouts** (truncation hinting, all methods in Fig. 18). **Left:** accuracy versus hint level H, one curve per model, colored by ground-truth rank at $H = 0$ from light (lowest) to dark (highest). **Middle:** Kendall's τ between the ground-truth ranking and the ranking from one rollout per problem at each H. Bands are ± 1 std over simulated single-rollout runs. **Right:** Kendall's τ against the budget of k rollouts per problem for **avg@**k, the **hint-prior ranking** (Eq. (2)), the single-level rankings of the middle panel (**semi-transparent lines**), and the **log-loss baseline**.

5 Ranking Models by Benchmark Accuracy

We explore whether hinting can help with sample-efficient model ranking. We focus on models with similar accuracies near zero, where tiny accuracy gaps make an unhinted ranking need many rollouts. Hinting widens these gaps while approximately retaining the rank order so that fewer rollouts recover it. While §4 tested whether hinting can help predict performance of models across a wide range of capabilities, here we test whether hinting can tell apart models of nearly identical capability.

Setting. We call models *near-zero* if all their accuracies lie within a few percentage points of the floor and *nearly indistinguishable* if their accuracies span at most two percentage points. On AIME and HLE, we build such a set of seven models and obtain a ground-truth ranking by drawing a large number of samples (see §G.1). We report results for the truncation hinting method in the main text, which we find works best for ranking (see §G.3 for other methods). We evaluate a ranking by Kendall's τ (Kendall, 1938) with the ground-truth ranking, where $\tau = 1$ is perfect agreement and $\tau = 0$ is chance. Each ranking method receives a budget of k rollouts per problem per model, and we report τ averaged over 1,000 simulated runs at that budget. We compare three rankings: *avg@*k, which averages the k unhinted rollouts; a *log-loss baseline*, which needs no rollouts and ranks models by their teacher-forced log loss on the reference solutions; and our *hint-prior ranking*. The hint-prior ranking spends one rollout per problem on a hinted benchmark that mixes hint levels across problems and the remaining $k - 1$ on the unhinted one. Writing $\widehat{\mathrm{Acc}}_1(M, \bar{H})$ and $\widehat{\mathrm{Acc}}_{k-1}(M, 0)$ for the resulting hinted and unhinted scores and $z[\cdot]$ for the z-score across models, it ranks models by

$$\hat{s}(M,k) = (1-\alpha_k)\, z\Big[\widehat{\mathrm{Acc}}_1(M,\bar{H})\Big] + \alpha_k\, z\Big[\widehat{\mathrm{Acc}}_{k-1}(M,0)\Big], \qquad \alpha_k = \frac{k-1}{k-1+\lambda}, \tag{2}$$

so the hinted score is worth λ unhinted rollouts per problem and at $k = 1$ the ranking is purely hinted. We set $\lambda = 8$ throughout. §G.2 gives the full construction and the choice of λ.

Hinting preserves ranking. In Fig. 7 (left), hinting spreads the models' accuracies apart while largely preserving their unhinted order. Darker curves indicate higher performance on the original benchmark ($H = 0$), and we observe that with increasing H, curves roughly stay in color order. Not every hint level preserves it equally well, however. Fig. 7 (middle) shows how well one hinted rollout per problem recovers the ranking at each level, and the best level differs between benchmarks: Kendall's τ peaks at $H = 0.8$ on AIME and drops sharply at $H = 1$, while on HLE it rises through $H = 1$. The hint-prior ranking avoids tuning H by using all levels.

Benchmark Hinting enables sample-efficient estimation of the ground-truth ranking. At small budgets, the hint-prior ranking is far closer to the ground truth than avg@k, the ranking by unhinted

accuracy alone (Fig. 7, right). At one rollout per problem, it leads avg@k by 0.33 in Kendall's τ on AIME and 0.26 on HLE, and avg@k requires about 5 rollouts per problem on AIME and 6 on HLE to match it. The lead narrows as the budget grows and the weight shifts to the unhinted score. The two converge at 128 rollouts per problem on AIME and 64 on HLE, the largest budgets we evaluate, and at no budget is the hint-prior ranking behind avg@k by more than 0.01 in τ. The convergence holds by construction: α_k approaches one as the budget grows, so the hint-prior ranking becomes avg@k at large budgets whether or not the hinted ranking is correct. The log-loss baseline uses a continuous metric rather than averaging rollouts, and does worse than both sampling methods. It reaches $\tau = 0.33$ on AIME, a level avg@k needs about two rollouts per problem to match and the hint-prior ranking below one, and $\tau = -0.05$ on HLE, below chance.

6 Related Work

Downstream Scaling Laws. Downstream scaling laws extrapolate benchmark accuracy from a ladder of small models to a larger target, but where the smallest models score at chance the fit is unconstrained and often unreliable (Lourie et al., 2025). Prior works have attempted to fit accuracy only on models above chance, either directly (Krajewski et al., 2025) or through a pretraining-loss proxy (Chen et al., 2025; Bhagia et al., 2025; Gadre et al., 2024). Others replace accuracy with a continuous metric such as the likelihood of the correct answer (Schaeffer et al., 2023; Magnusson et al., 2025; Olmo et al., 2026), which has signal at small scale but is a less useful quantity to predict. Closest to our work are methods that extract signal from models at chance by sampling each problem many times (Hu et al., 2024; Krajewski et al., 2025) or by finetuning on task data (Snell et al., 2024). Benchmark Hinting belongs to this group but differs in requiring no additional inference or fine-tuning, instead using prompt- and logit-based methods to boost signal.

Ranking models cheaply. Reducing the cost of model evaluation and ranking is a well-studied problem, with some works focusing on evaluating fewer problems (Zhang et al., 2026) and others replacing rollouts with teacher-forced statistics on reference solutions (Olmo et al., 2026; Patel et al., 2026). In §5 we instead reduce the rollouts per problem by using hinted performance as a biased prior that improves sample efficiency.

Varying problem difficulty. Some prior works have explored varying problem difficulty in a controlled way. Jones (2021) scales the difficulty of board games and finds that the performance degrades predictably as the game gets harder, so scaling laws extrapolate across problem difficulty. Closer to our truncation method, Qu et al. (2026) and Setlur et al. (2026) place a prefix of a correct solution in a language model's context to form a curriculum during reinforcement learning. However, our work is the first to leverage these ideas for language model evaluations.

7 Discussion

This work initiates the study of hinted scaling laws and posits that jointly modeling the effects of model capability and hint level on benchmark accuracy is more predictive than capability alone. We consider two prediction problems: forecasting benchmark accuracy across a model family and ranking models from different families.

A limitation of our study on forecasting is the scarcity of open-source scaling ladders. Among the five model families we consider, only Delphi is explicitly a scaling ladder (i.e., trained on the same data recipe). Thus, the extent to which our forecasting errors are attributable to shortcomings of our methods versus artifacts of these model families is unclear. Certainly, our simple joint law is an imperfect model of the true joint effect of hint level and model capability on accuracy.

Developing sample-efficient methods for ranking models on benchmarks is particularly important in view of the substantial cost of running modern agentic benchmarks. For example, a single run of UK AISI's cyber-range evaluation costs roughly $3000–$5000 per rollout per frontier model (AI Security Institute, 2026). Safety and red-teaming benchmarks are often expensive because many rollouts are required to estimate long-tail risks (Angell et al., 2026; Jones et al., 2025). Designing new hinting methods that work well for these benchmark types is an important future direction.

AI USE STATEMENT

In this work, we used generative AI tools for generating synthetic data sets (HLE and AIME solutions were generated by AI models; HLE is graded by LLM judge), proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, implementing methods, interpreting results, supporting qualitative and thematic data analysis and formulating mathematical claims. We have not used generative AI tools for helping to develop theoretical models or conceptual frameworks, assisting with translation and cleaning and reformating a dataset. Additionally, we used generative AI tools for creating or modifying scientific figures or images, creating or editing software code, creation of artifacts, assisting writing, editing a research paper to improve readability and summarizing or analysing existing literature. We have reviewed all AI-assisted work. We have read over all AI writing to make sure of correctness and sampled the generated AIME and HLE solutions (these were also graded as correct by, for AIME, a deterministic grader and for HLE an LLM judge). We have ensure any artifacts created are based on actual research data and checked code. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses only public benchmarks and openly released model weights, under their respective licenses and involves no human subjects or personal data. A hinted accuracy is meaningful only together with its hint level and must not be reported as a score on the original benchmark. Hinted scaling laws forecast when a capability will appear at a larger scale. We see this mainly as a benefit for safety evaluation, where the capabilities of most concern are those on which small proxy models score zero, and the method measures models without improving them.

REPRODUCIBILITY STATEMENT

We will release the hinted benchmarks, including the reference solutions, the hints at every level and every hinting method, and all rollouts with their grades, together with the code that generates hints, runs models, grades outputs, fits the scaling laws and produces every figure in this paper. §A describes the benchmarks and the HLE subset, §B the generation and verification of the reference solutions, §E the fitting procedure and baselines, and §G the model sets, ground truth and simulation behind the ranking results.

REFERENCES

AI Security Institute. How fast is autonomous ai cyber capability advancing? https://www.aisi.gov.uk/blog/how-fast-is-autonomous-ai-cyber-capability-advancing, 2026. Accessed: 2026-09-01.

Rico Angell, Raghav Singhal, Zachary Horvitz, Zhou Yu, Rajesh Ranganath, Kathleen McKeown, and He He. Estimating tail risks in language model output distributions. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2604.22167.

Akshita Bhagia, Jiacheng Liu, Alexander Wettig, David Heineman, Oyvind Tafjord, Ananya Harsh Jha, Luca Soldaini, Noah A. Smith, Dirk Groeneveld, Pang Wei Koh, Jesse Dodge, and Hannaneh Hajishirzi. Establishing task scaling laws via compute-efficient model ladders, 2025. URL https://arxiv.org/abs/2412.04403.

Yangyi Chen, Binxuan Huang, Yifan Gao, Zhengyang Wang, Jingfeng Yang, and Heng Ji. Scaling laws for predicting downstream performance in llms, 2025. URL https://arxiv.org/abs/2410.08527.

Samir Yitzhak Gadre, Georgios Smyrnis, Vaishaal Shankar, Suchin Gururangan, Mitchell Wortsman, Rulin Shao, Jean Mercat, Alex Fang, Jeffrey Li, Sedrick Keh, Rui Xin, Marianna Nezhurina, Igor Vasiljevic, Jenia Jitsev, Luca Soldaini, Alexandros G. Dimakis, Gabriel Ilharco, Pang Wei Koh, Shuran Song, Thomas Kollar, Yair Carmon, Achal Dave, Reinhard Heckel, Niklas Muennighoff, and Ludwig Schmidt. Language models scale reliably with over-training and on downstream tasks, 2024. URL https://arxiv.org/abs/2403.08540.

Will Held. Scaling laws that extrapolate 300× past the fit. https://www.openathena.ai/blog/delphi/, may 2026. Open Athena Blog.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022. URL https://arxiv.org/abs/2203.15556.

Shengding Hu, Xin Liu, Xu Han, Xinrong Zhang, Chaoqun He, Weilin Zhao, Yankai Lin, Ning Ding, Zebin Ou, Guoyang Zeng, Zhiyuan Liu, and Maosong Sun. Predicting emergent abilities with infinite resolution evaluation, 2024. URL https://arxiv.org/abs/2310.03262.

Andy L. Jones. Scaling scaling laws with board games, 2021. URL https://arxiv.org/abs/2104.03113.

Erik Jones, Meg Tong, Jesse Mu, Mohammed Mahfoud, Jan Leike, Roger Grosse, Jared Kaplan, William Fithian, Ethan Perez, and Mrinank Sharma. Forecasting rare language model behaviors, 2025. URL https://arxiv.org/abs/2502.16797.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938. ISSN 00063444. URL http://www.jstor.org/stable/2332226.

Jakub Krajewski, Amitis Shidani, Dan Busbridge, Sam Wiseman, and Jason Ramapuram. Revisiting the scaling properties of downstream metrics in large language model training, 2025. URL https://arxiv.org/abs/2512.08894.

Hong Liu, Sang Michael Xie, Zhiyuan Li, and Tengyu Ma. Same pre-training loss, better downstream: Implicit bias matters for language models. In *International Conference on Machine Learning*, pp. 22188–22214. PMLR, 2023.

Nicholas Lourie, Michael Y. Hu, and Kyunghyun Cho. Scaling laws are unreliable for downstream tasks: A reality check, 2025. URL https://arxiv.org/abs/2507.00885.

Ian Magnusson, Nguyen Tai, Ben Bogin, David Heineman, Jena D. Hwang, Luca Soldaini, Akshita Bhagia, Jiacheng Liu, Dirk Groeneveld, Oyvind Tafjord, Noah A. Smith, Pang Wei Koh, and Jesse Dodge. Datadecide: How to predict best pretraining data with small experiments, 2025. URL https://arxiv.org/abs/2504.11393.

Team Olmo, :, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Morrison, Jake Poznanski, Kyle Lo, Luca Soldaini, Matt Jordan, Mayee Chen, Michael Noukhovitch, Nathan Lambert, Pete Walsh, Pradeep Dasigi, Robert Berry, Saumya Malik, Saurabh Shah, Scott Geng, Shane Arora, Shashank Gupta, Taira Anderson, Teng Xiao, Tyler Murray, Tyler Romero, Victoria Graf, Akari Asai, Akshita Bhagia, Alexander Wettig, Alisa Liu, Aman Rangapur, Chloe Anastasiades, Costa Huang, Dustin Schwenk, Harsh Trivedi, Ian Magnusson, Jaron Lochner, Jiacheng Liu, Lester James V. Miranda, Maarten Sap, Malia Morgan, Michael Schmitz, Michal Guerquin, Michael Wilson, Regan Huff, Ronan Le Bras, Rui Xin, Rulin Shao, Sam Skjonsberg, Shannon Zejiang Shen, Shuyue Stella Li, Tucker Wilde, Valentina Pyatkin, Will Merrill, Yapei Chang, Yuling Gu, Zhiyuan Zeng, Ashish Sabharwal, Luke Zettlemoyer, Pang Wei Koh, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. Olmo 3, 2026. URL https://arxiv.org/abs/2512.13961.

Arkil Patel, Siva Reddy, Marius Mosbach, and Dzmitry Bahdanau. Forecasting downstream performance of llms with proxy metrics, 2026. URL https://arxiv.org/abs/2605.18607.

Yuxiao Qu, Amrith Setlur, Virginia Smith, Ruslan Salakhutdinov, and Aviral Kumar. Pope: Learning to reason on hard problems via privileged on-policy exploration, 2026. URL https://arxiv.org/abs/2601.18779.

Yangjun Ruan, Chris J. Maddison, and Tatsunori Hashimoto. Observational scaling laws and the predictability of language model performance, 2024. URL https://arxiv.org/abs/2405.10938.

Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. Are emergent abilities of large language models a mirage?, 2023. URL https://arxiv.org/abs/2304.15004.

Amrith Setlur, Zijian Wang, Andrew Cohen, Paria Rashidinejad, and Sang Michael Xie. Reuse your flops: Scaling rl on hard problems by conditioning on very off-policy prefixes, 2026. URL https://arxiv.org/abs/2601.18795.

Charlie Snell, Eric Wallace, Dan Klein, and Sergey Levine. Predicting emergent capabilities by finetuning, 2024. URL https://arxiv.org/abs/2411.16035.

Jacob Mitchell Springer, Sachin Goyal, Kaiyue Wen, Tanishq Kumar, Xiang Yue, Sadhika Malladi, Graham Neubig, and Aditi Raghunathan. Overtrained language models are harder to fine-tune. *arXiv preprint arXiv:2503.19206*, 2025.

Ishaan Watts, Catherine Li, Sachin Goyal, Jacob Mitchell Springer, and Aditi Raghunathan. Sharpness-aware pretraining mitigates catastrophic forgetting. *arXiv preprint arXiv:2605.02105*, 2026.

Guanhua Zhang, Florian E. Dorner, and Moritz Hardt. How benchmark prediction from fewer data misses the mark, 2026. URL https://arxiv.org/abs/2506.07673.

A BENCHMARK DETAILS

AIME. We use the 60 problems of the 2025 and 2026 AIME I and II exams taken from MathArena. Every answer is an integer between 0 and 999. We prompt models to put their final answer in `\boxed{}`, extract the final answer from the response with a rule-based parser, and mark the response correct if it equals the official answer.

HLE. From the 2,500 questions of Humanity's Last Exam we use a 176-question subset. First, we only consider questions without images. Of these, we keep a question if (1) its answer is free-form (`exactMatch`) rather than multiple-choice, (2) we can generate a correct reference solution for it (§B), and (3) it does not hand the model a set of candidate answers. For (3) we manually removed 14 free-form questions that ask a yes/no question, enumerate options in the question text, or ask "how many of the following". Following the HLE protocol, we judge correctness with an LLM. We use the HLE judge prompt with its confidence field removed, since we do not ask models for a confidence score, and GPT-5.4 nano (low reasoning effort) as the judge. The judge sees the question, the ground-truth answer and the final 1,500 characters of the model's response, and returns the extracted final answer and a yes/no verdict.

GSM8K. We use the first 256 problems of the GSM8K test split. The reference solutions are the step-by-step solutions that come with the benchmark.

B REFERENCE SOLUTIONS FOR AIME AND HLE

Benchmark Hinting methods sometimes require a correct solution to every problem. For example, both masking and truncation hinting give part of the solution to the model. Such a correct solution should not state the final answer, otherwise the model could copy it without building on the hint. This is especially relevant for masking, where the hint level H would otherwise just be the probability of revealing the answer to the model. Thus, we make sure that the final answer is not in the correct solution.

We build the solutions in two stages. First we sample full solutions from a frontier model and keep only ones that are deemed correct by a grader. Second, we make a model rewrite each correct solution so that it does not include the final answer and stops just before it. We collect ten reference solutions per problem on both benchmarks, and the hinted rollouts of a problem cycle through them.

Stage 1: sampling correct solutions. We prompt a frontier model with the problem and keep a sample only if it is graded correct. On AIME a sample is correct if the extracted integer matches the official answer; on HLE we apply the LLM judge of §A with o3-mini (low reasoning effort) as the judge. We resample until every problem has ten correct solutions. On AIME the solver is Claude Opus 4.6 with extended thinking off, at medium or high effort. The remaining rollouts of a few problems on which Opus 4.6 rarely succeeded were sampled with GPT-5.4 with medium thinking. On HLE the solver is GPT-5.5 with medium thinking.

Stage 2: removing the answer. After collecting correct solutions, we make a model rewrite them so they do not contain the problem answer. This second model is given the problem, the solution and the answer, and is instructed to stop before the final step and to never state the answer. On AIME the rewriter is Claude Opus 4.6 with extended thinking off. On HLE the rewriter is Claude Opus 4.7 with medium thinking. The rewritten solutions are the reference solutions used by all three Benchmark Hinting methods.

Checking for leaked answers. To check that the rewritten solutions do not state the final answer, we search each one for its problem's answer string with a regular expression. The string occurs in 19 of the 600 AIME reference solutions and 165 of the 1,760 HLE ones. A manual spot check shows that most of these are not true leaks. The answer string usually appears in the explanation without being marked as the answer, e.g. as a quantity given in the problem, an intermediate value, part of a larger number, or a symbol used in the derivation.

C METHOD DETAILS

C.1 LOGIT AVERAGING

§2.1 describes logit averaging over the full vocabulary. In practice each model reports only its k largest logits at each step, and we mix over the union of the two sets. We run the evaluated model and the teacher as two separate vLLM engines that decode together. At every step each engine computes its raw next-token logits and reports the top k, giving token index sets $\mathcal{T}_{\text{model}}$ and $\mathcal{T}_{\text{teacher}}$ together with their logits. We then form for every $j \in \mathcal{T}_{\text{model}} \cup \mathcal{T}_{\text{teacher}}$,

$$s_H(x, y_{<t})_j = (1 - H)\, \tilde{z}_{\text{model}}(x, y_{<t})_j + H\, \tilde{z}_{\text{teacher}}(x, y_{<t})_j,$$

where $\tilde{z}_{\text{model}}(x, y_{<t})_j$ is the reported logit when $j \in \mathcal{T}_{\text{model}}$, and otherwise the smallest logit the model reported, $\min_{i \in \mathcal{T}_{\text{model}}} z_{\text{model}}(x, y_{<t})_i$, and likewise for the teacher. We then sample $y_t \sim \text{softmax}(s_H/T)$ over the union at the evaluated model's temperature T. We use $k = 128$ for the post-trained families and $k = 1000$ for the Delphi models, whose standard sampling uses a top-k of 1000.

D HINTING AND CAPABILITY SCALING: ALL FAMILIES

§3 shows the Delphi ladder and Qwen3.5 on HLE. Here we show the same analyses for the remaining (benchmark, family) pairs: Qwen2.5, Qwen3 and Gemma-3 on both AIME and HLE. Fig. 8 gives the hint and capability marginals, with each marginal fitted alone as in Fig. 2, and Fig. 9 gives the empirical iso-accuracy contours, as in Fig. 3. The same patterns hold. Accuracy rises smoothly with the hint level for every model, the ordering of each family is preserved, and the contours are smooth. We did not run Qwen3.5 on AIME.

E PREDICTING BENCHMARK PERFORMANCE

Figs. 10 and 11 show the joint-law fits and the forecasts for the (benchmark, family) pairs not reported in §4: Qwen3 on AIME, and Qwen2.5 and Gemma-3 on HLE. We did not run Qwen3.5 on AIME.

Fig. 12 compares the empirical iso-accuracy contours of Fig. 3 with the contours implied by the joint law of Eq. (1). Under that law every contour is a straight line with the same slope $-\beta_1/\beta_2$, so the lower half of the figure shows what the surface would look like if the additive form held exactly. For Qwen3.5 on HLE the empirical contours are close to the implied ones for all three Benchmark Hinting methods. For the Delphi ladder on GSM8K the empirical contours are flat across the small models and turn steep at the largest two, a shape that no single slope can reproduce, so the implied contours cut through the low-accuracy corner where the data are flat. This is the discrepancy behind the Delphi logit averaging hint marginals in Fig. 4, and the reason we describe the joint law as an approximation.

E.1 TWO STAGE SCALING LAW BASELINE

A natural alternative to Benchmark Hinting is to forecast through an intermediate *soft* metric rather than through a hinted benchmark. The soft metric is a continuous quantity that keeps moving while accuracy is still pinned at the floor, and the forecast is a chain of two fits, compute to the soft metric, then the soft metric to accuracy. Held (2026) use exactly this procedure on the Delphi ladder, combining a compute scaling law with an *observational* scaling law (Ruan et al., 2024). We test this method here as a baseline.

Following Held (2026), we use bits-per-byte on the GSM8K solution trace as the soft metric averaged over problems. We use the same standard 5-shot prompt. As a check on our implementation, our measurements track those reported by Held (2026) for the same ladder closely, differing by 0.84% on average and 2.09% at worst across all ten rungs, despite differences in problem subset and prompt details.

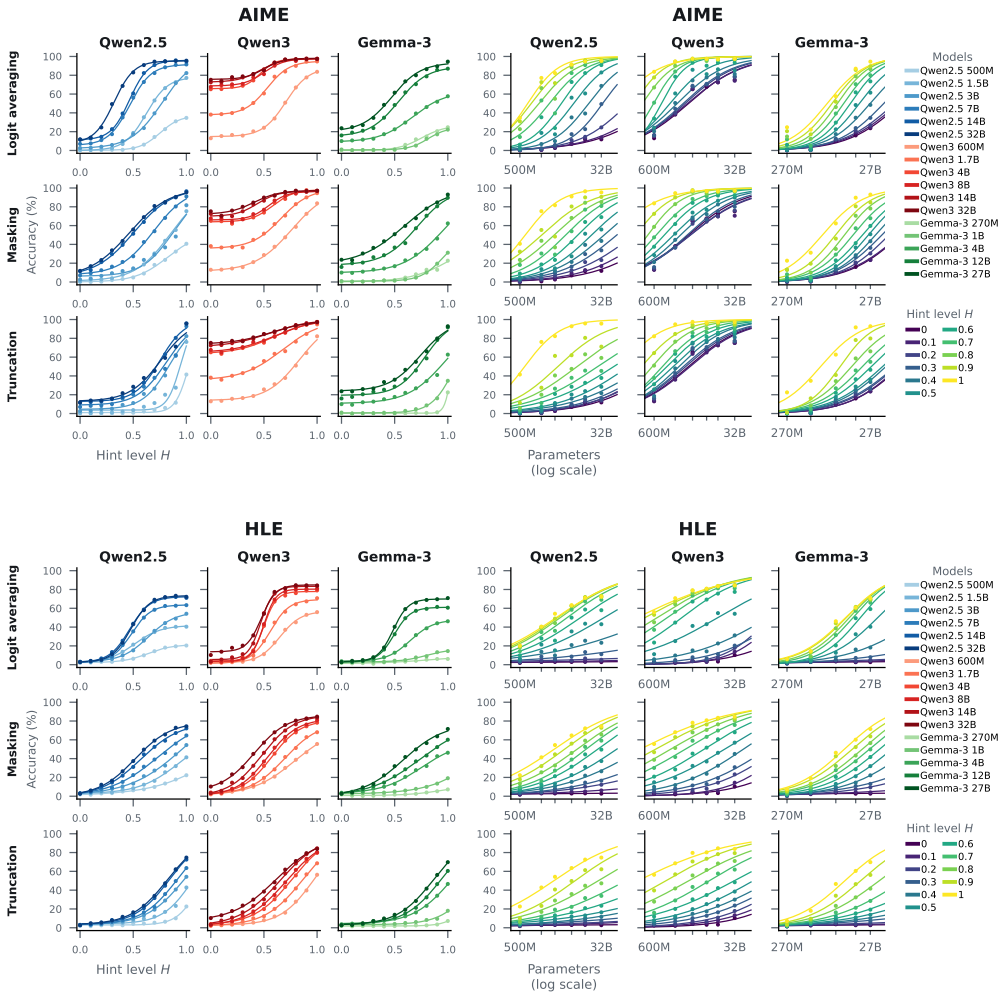

Figure 8: **Hint and capability marginals for the remaining families.** As Fig. 2, for Qwen2.5, Qwen3 and Gemma-3 on AIME (top) and HLE (bottom). Rows are the three Benchmark Hinting methods. **Left:** accuracy against hint level H, one curve per model, darker is larger. **Right:** accuracy against model scale, one curve per hint level. Each curve is a sigmoid fitted to that model or that level alone, with free asymptotes in H and asymptotes pinned to 0 and 1 in scale.

We then fit two scaling laws and compose them. The first maps pretraining FLOPs C to bits-per-byte with a power law, and the second maps bits-per-byte to accuracy with a sigmoid. Concretely:

$$\mathrm{BPB}(C) = L_\infty + A\,C^{-\alpha}, \tag{3}$$

$$\mathrm{Acc}(M, 0) = \sigma(b_1 \mathrm{BPB}(M) + b_0)\,. \tag{4}$$

We consider two different ways of fitting these laws and creating a forecast. The most comparable to our results in Fig. 4 is to fit both on the t smallest models in the ladder, and evaluate on the held-out larger models. The results of doing this are shown on the left of Fig. 14, with the absolute error of the prediction shown in blue on the bottom plot. We find this method does *very poorly*, predicting essentially a flat line at 0, resulting in a roughly constant absolute error of 17–22 accuracy points. This makes perfect sense because the second stage of the fit in Eq. (4) is simply fitting a sigmoid to a row of near-zero accuracy values, the exact same problem as the $H = 0$ sigmoid we use as a baseline in the main paper.

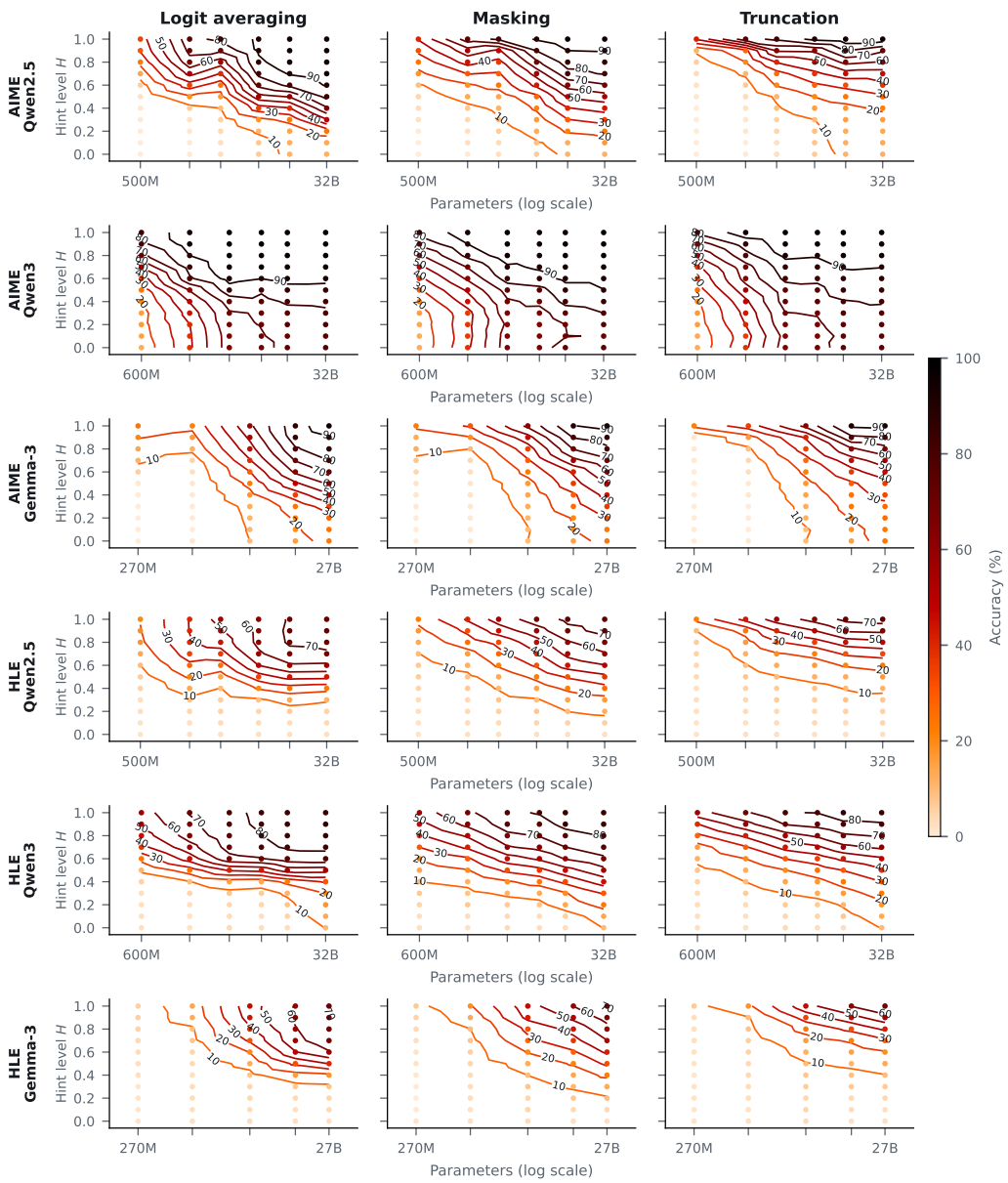

Figure 9: **Iso-accuracy contours for the remaining families.** As Fig. 3, for Qwen2.5, Qwen3 and Gemma-3 on AIME and HLE. Columns are the three Benchmark Hinting methods.

We also consider a second setting used by Held (2026). In this case we imagine we have access to a set of stronger *reference* pretrained models, which already exceed the accuracy of the largest model we plan on scaling to. It is important to note that this assumption is *not realistic* if you are running a frontier training run beyond the scale of any publicly available base model. However, let us assume this is possible, then you can measure these models' benchmark accuracies and bits-per-byte, and use them to anchor the bits-per-byte to accuracy sigmoid.

We show results for this reference-model fit on the right of Fig. 14. We find that the sigmoid is indeed fitted well, however with enough held-out models the compute to bits-per-byte scaling law ends up being inaccurate, leading also to poor predictions overall. It is worth emphasizing that the downstream accuracy prediction is *very* sensitive to even small errors in the bits-per-byte prediction

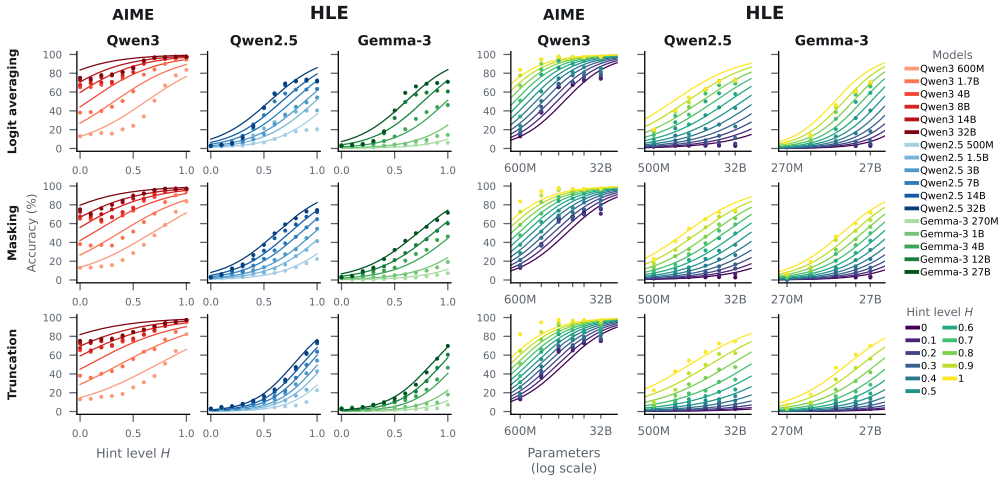

Figure 10: As Fig. 5, for Qwen3 on AIME and Qwen2.5 and Gemma-3 on HLE.

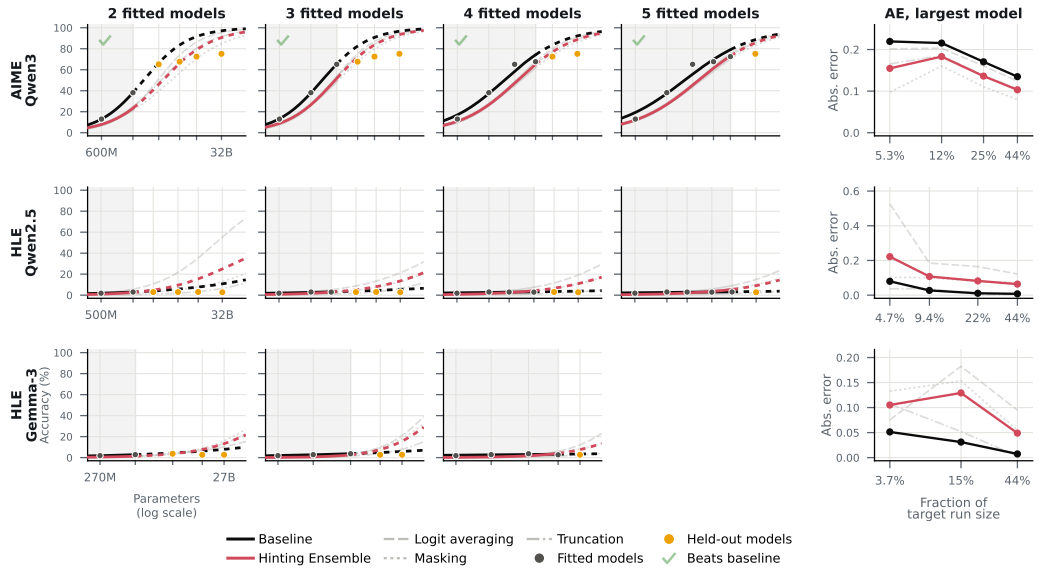

Figure 11: As Fig. 6, for Qwen3 on AIME and Qwen2.5 and Gemma-3 on HLE.

(see 3 held-out models of Fig. 14). On the other hand, the model we wish to predict, $C = 10^{23}$, has only just emerged on the bits-per-byte to accuracy sigmoid, and so it makes sense that small bits-per-byte errors would have large accuracy errors as the true 10^{23} model sits close to the steepest point of the sigmoid.

Finally, as expected, the bits-per-byte of the post-trained model ladders scales much less cleanly than the Delphi ladder, as shown in Fig. 13, making the method not feasible for these ladders.

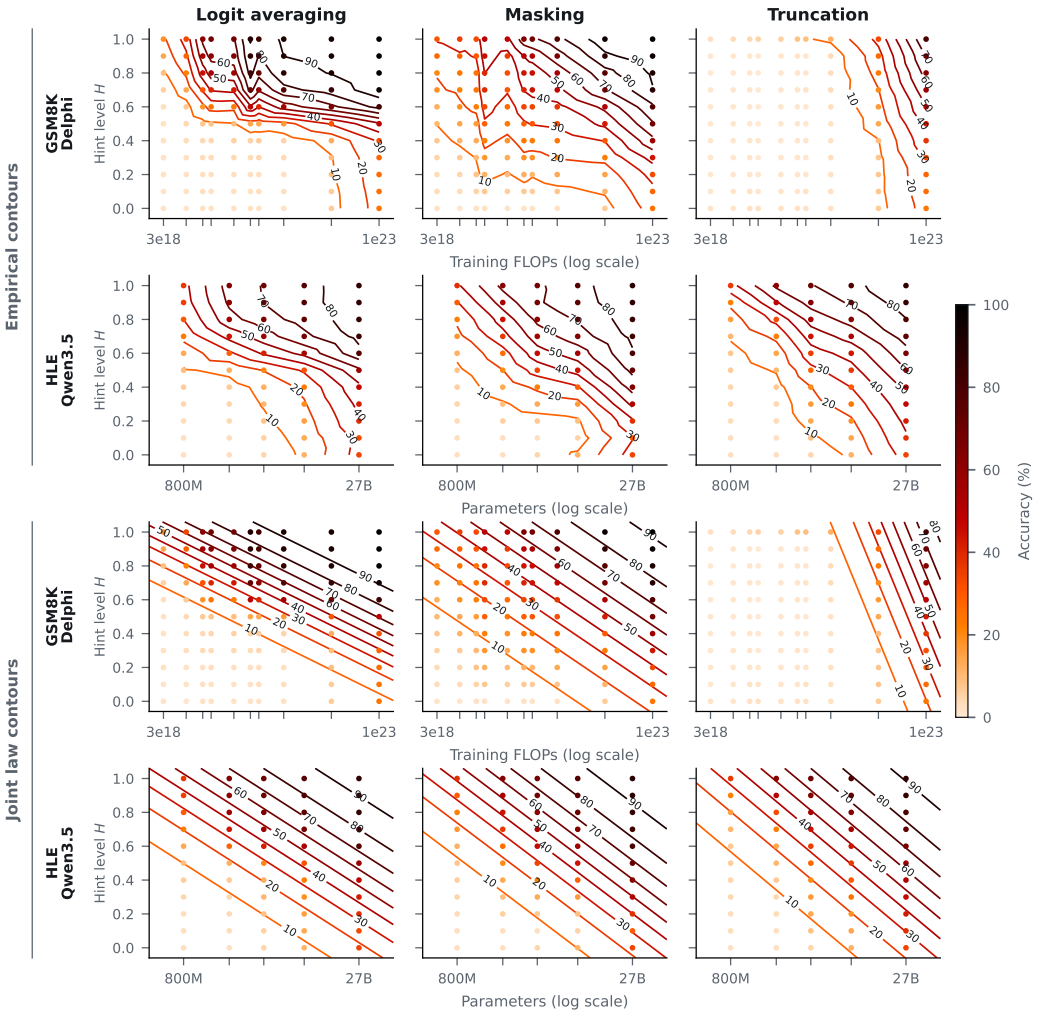

Figure 12: **Empirical against joint-law iso-accuracy contours. Top two rows:** Fig. 3, the Delphi ladder on GSM8K and the Qwen3.5 family on HLE with contours interpolated from the measured cells. **Bottom two rows:** the same cells with the contours of the joint law of Eq. (1) fitted to each panel, which are straight parallel lines of slope $-\beta_1/\beta_2$. Columns are the three Benchmark Hinting methods. Each dot is one measured cell colored by accuracy.

E.2 PASS@k AS A HINTING METHOD

Here we consider pass@k as a hinting method. *Pass*@k marks a problem as solved if any of k independently sampled attempts is correct, so $k = 1$ is the ordinary evaluation and accuracy rises monotonically in k.

It is easy to see that pass@k is a valid Benchmark Hinting method. It is indexed by a scalar, it returns the unmodified benchmark at one end of its range, and it raises accuracy monotonically as it is turned up. We therefore treat it as one more Benchmark Hinting method, with hint level $H = \log_2 k$ so that $H = 0$ is the unmodified benchmark and each additional unit of H doubles the sampling budget. We then fit the same additive joint law of Eq. (1) over the resulting grid of (scale, hint) pairs and forecast exactly as before, by evaluating the fitted law's $H = 0$ slice at the held-out scales.

Fig. 15 shows the result. The joint law describes the pass@k marginals well. In terms of forecasting, the method generally improves on the baseline but performs particularly poorly on the Delphi ladder. It also underperforms our main ensembling method.

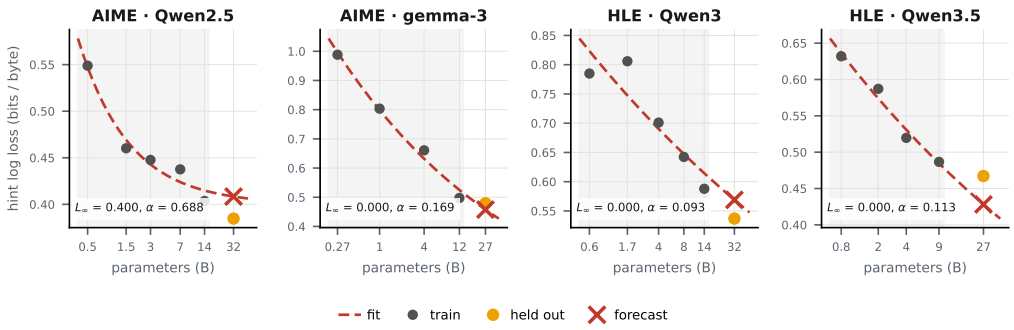

Figure 13: **Bits-per-byte against parameter count on the post-trained families.** Each panel holds out the largest model and fits Eq. (3) to the rest (the **shaded** region), then reads the fit at the held-out model's parameter count. ● are fitted models, ● the held-out model and **crosses** the forecast. Fitted L_∞ and α are annotated per panel; L_∞ reaches its lower bound of zero in three of the four families, indicating the floor is not identified at this number of points.

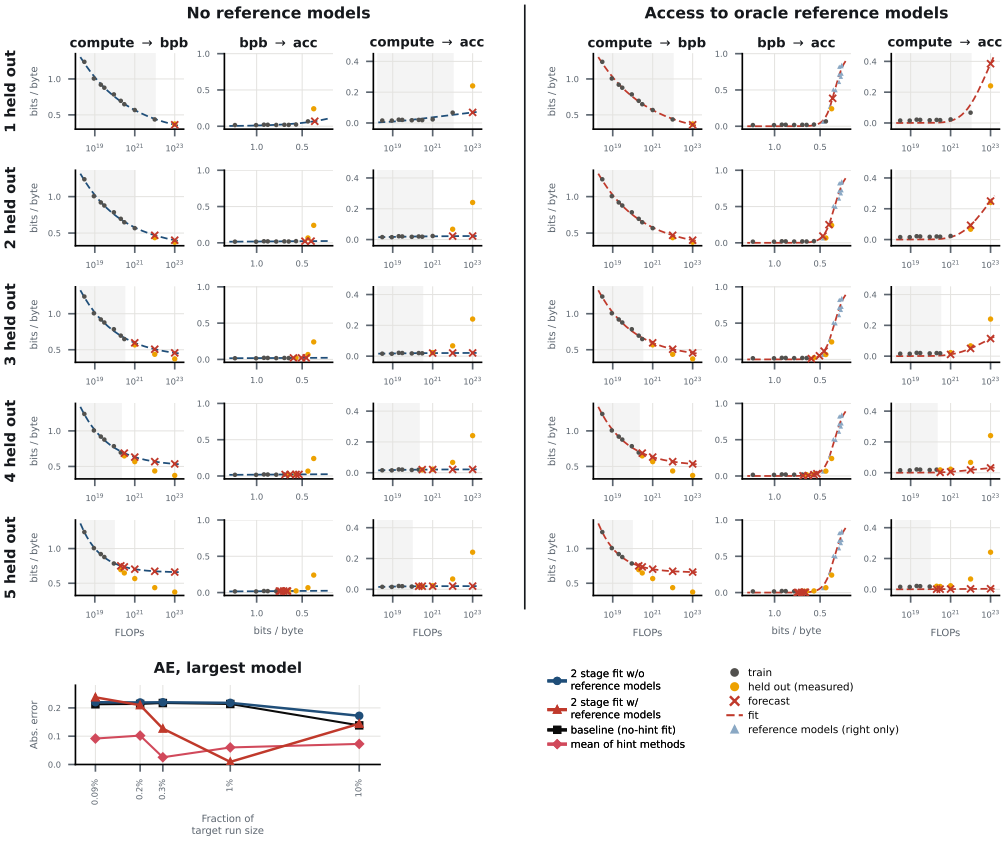

Figure 14: **Two stage scaling law forecasts on the Delphi ladder (GSM8K).** Each row is fitted on the smallest models only (the **shaded** region) and forecasts the held-out larger ones. **Columns within a block:** the compute-to-BPB power law of Eq. (3); the BPB-to-accuracy sigmoid of Eq. (4), with the x axis running high to low so the curve rises left to right; and the two composed, giving accuracy implied at any compute. ● are fitted models and ● held-out models; **crosses** are forecasts, which use only the held-out model's compute. **Left block:** the sigmoid is fitted on the fitted models alone. **Right block:** the eleven external reference models of Held (2026) (triangles) are added to its fitting data. **Bottom:** absolute error on the largest model against the size of the largest fitted model as a fraction of the target's, with the unhinted baseline and the mean of the Benchmark Hinting methods from Fig. 4 for reference.

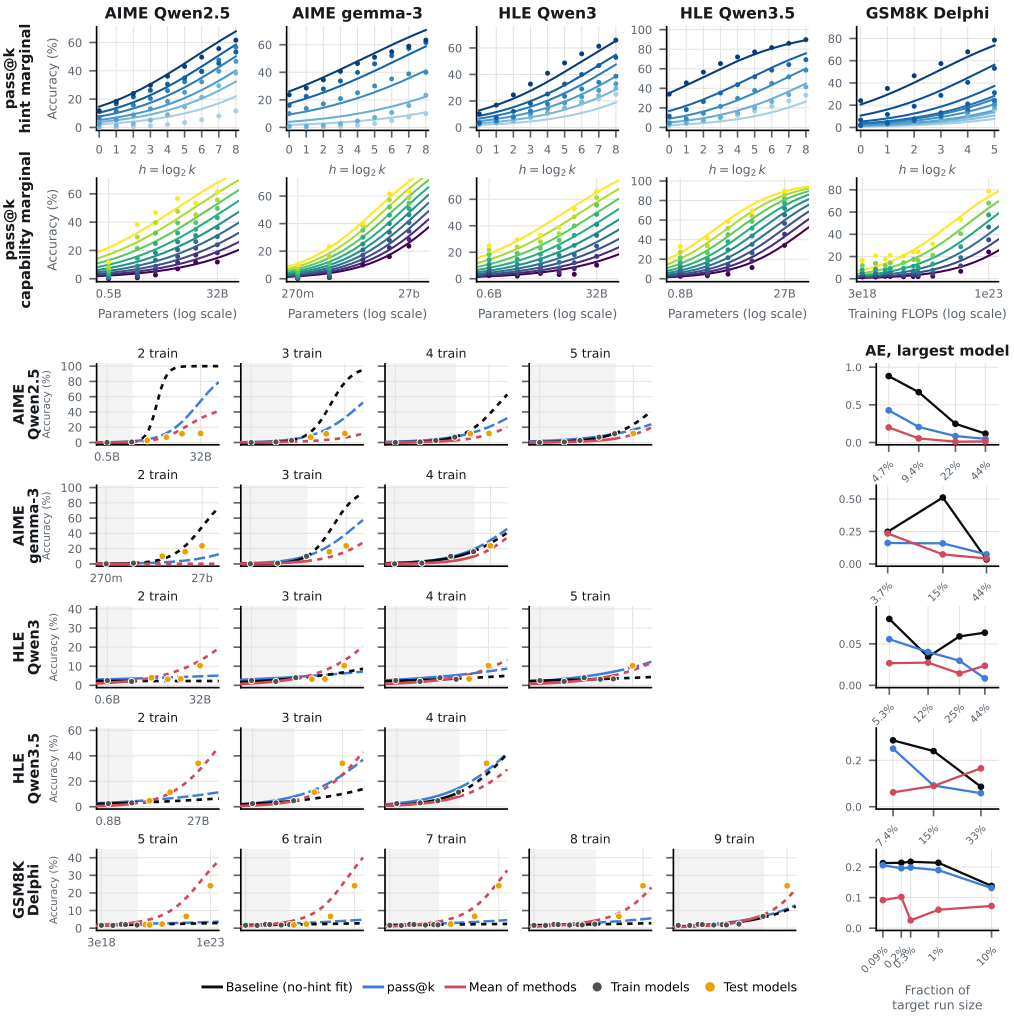

Figure 15: **Pass@k as a Benchmark Hinting method. Top two rows:** the joint law of Eq. (1) fitted to the pass@k grid of each (benchmark, family) pair, drawn as its hint marginals (accuracy against $H = \log_2 k$, one line per model, darker with scale) and its capability marginals (accuracy against scale, one line per hint level, brighter with H). **Bottom:** the same law refitted on the t smallest models only (the shaded region) and evaluated at $H = 0$ on the held-out larger ones. We compare against the ensemble of the three Benchmark Hinting methods in the main paper and the regular sigmoid fitted to the $H = 0$ accuracies.

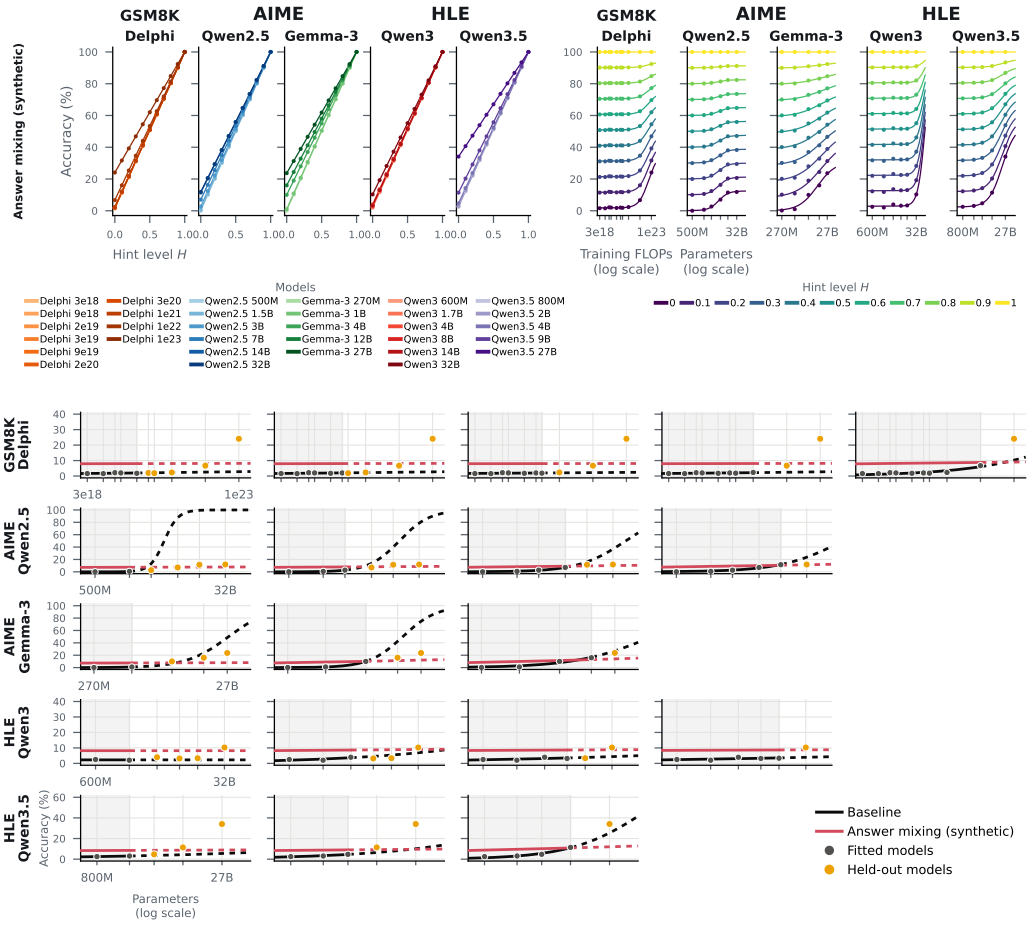

Figure 16: **Answer mixing carries no information for forecasting. Top:** hint and capability marginals of answer mixing, generated from each family's unhinted accuracies as $\mathrm{Acc}(M, h) = h + (1 - h)\,\mathrm{Acc}(M, 0)$. Each hint marginal is fitted with a straight line and each capability marginal with a sigmoid with free asymptotes. **Bottom:** forecasts of the largest model's unhinted accuracy from the joint law of Eq. (1) fitted to the answer-mixing cells of the smallest models (red), against the baseline sigmoid fitted to the unhinted scores alone (black). The shaded region spans the fitted models, grey dots are fitted models and orange dots held-out models.

F When can hinting inform a forecast?

Not every way of making a benchmark easier helps with forecasting. As a control, consider *answer mixing*. With probability H the model's answer is marked correct regardless of its content, and otherwise the model's own answer is graded as usual. Its accuracy is $\mathrm{Acc}(M, h) = h + (1 - h)\,\mathrm{Acc}(M, 0)$, so every hint marginal is a straight line from the model's unhinted score to 1. In Fig. 16 we show empirically how this method performs.

From Fig. 16 we see that the method performs terribly for forecasting. Intuitively this makes sense. Because the hint marginals only depend on the accuracy of the model at $H = 0$, the hinted curves of all models with $\mathrm{Acc}(M, 0) = 0$ perfectly overlap. This means that every set of pre-emergence models *produces the same forecast*, which clearly cannot be correct.

From this, we get better intuition as to what makes a good Benchmark Hinting method, *one that separates pre-emergence models according to the difference in their underlying capability*. Below we

formalize this property of overlapping $H = 0$ curves, and how such Benchmark Hinting methods cannot exceed the performance of simply predicting based on unhinted benchmark scores.

Setup. A ladder has models $M_1, \dots, M_t$ at scales $C_1 < \cdots < C_t$ and a target $M_\star$ at $C_\star > C_t$. Write $a_i(h) = \mathrm{Acc}(M_i, h)$ and $a_\star = \mathrm{Acc}(M_\star, 0)$. A forecaster observes a grid of hint levels $\mathcal{H}$ and the data

$$D = \{(C_i, h, a_i(h)) : i \le t,\ h \in \mathcal{H}\},$$

and outputs $\hat{a}_\star = f(D)$. The standard baseline sees only the unhinted data $D_0 = \{(C_i, a_i(0)) : i \le t\}$. We call a Benchmark Hinting method *unhinted-determined* if there is a fixed function g, the same for every model, such that

$$a_M(h) = g\big(h, a_M(0)\big) \quad \text{for all models } M \text{ and levels } h. \tag{5}$$

Answer mixing is unhinted-determined with $g(h, a) = h + (1 - h)a$.

Proposition 1. *Let the ladder, and with it D, D_0 and $a_\star$, be drawn from any distribution P. For an unhinted-determined method, every forecaster f of D has a forecaster f_0 of D_0 with $f_0(D_0) = f(D)$ on every ladder. Consequently, for any loss ℓ,*

$$\min_f \mathbb{E}_P\, \ell\big(f(D), a_\star\big) = \min_{f_0} \mathbb{E}_P\, \ell\big(f_0(D_0), a_\star\big),$$

and the same holds with the expectation replaced by the worst case over ladders.

Proof. By Eq. (5), $D = \Phi(D_0)$, where Φ replaces each $(C_i, a_i(0))$ by $\{(C_i, h, g(h, a_i(0)))\}_{h \in \mathcal{H}}$ and depends only on the known g and $\mathcal{H}$. Take $f_0 = f \circ \Phi$. Since f and f_0 give the same output on every ladder, they have the same loss on every ladder, hence the same expected loss under P and the same worst-case loss. So the best forecaster of D is no better than the best forecaster of D_0. Conversely D contains the unhinted cells (as $0 \in \mathcal{H}$), so it is no worse either. □

In particular, fitting the joint law of Eq. (1) to the hinted cells of an unhinted-determined method is just one particular forecaster of D_0. When every fitted model scores zero without hints, the unhinted data leave the target's accuracy completely unconstrained, so by Proposition 1 the hinted data of an unhinted-determined method leave it unconstrained too.

G Model ranking details

G.1 Model sets and ground truth

To be able to compare ranking methods, we need to have a ground truth to compare to. Here, we describe how we determine that ground truth and how we select the set of models to rank.

We use AIME and HLE as the benchmarks. On each benchmark we start from a pool of post-trained models from several families, 32 on AIME and 25 on HLE.

Collecting rollouts. To get an accurate ground-truth ranking, we compute an avg@k score with very large k. We first collect 128 and 192 unhinted rollouts per problem for AIME and HLE respectively. Then, we collect more rollouts for models that score near-zero: those below 1% on AIME, 14 models, receive 4,096 ground-truth rollouts per problem, and those below 3% on HLE, 16 models, receive 1,024. We use fewer rollouts on HLE because its grading requires a paid LLM judge. We only collect this deeper sample on part of the original model sets because it is compute-heavy and expensive to run. In both stages the rollouts we use for the ground truth are separate from the pool the ranking methods draw from. Deepening moves the estimates slightly, and one HLE model that was below 3% at 192 samples ends at 3.0%.

Creating a stable ranking and choosing a subset of models. Simply ranking models by avg@k with high k does not guarantee a ranking is correct. Thus, we only choose a subset of the models, chosen so every pair of models is certified at the 95% level with a Bonferroni correction over all pairs in the pool. We treat the benchmark problems as fixed, so the uncertainty is rollout sampling noise only, and form a confidence interval for each pair's accuracy difference from binomial standard errors computed from the observed per-problem accuracies. A pair is certified if the Bonferroni corrected

Table 2: Model pools for the ranking experiments. Models part of the set the ranking is evaluated on have a checkmark.

AIME

Model	Acc. (%)	Rollouts	Set
Qwen3 32B	75.091	128	
Gemma-4 31B	73.984	128	
Qwen3 14B	72.201	128	
Qwen3 30B-A3B	71.927	128	
Gemma-4 26B-A4B	71.003	128	
Qwen3 8B	68.086	128	
Qwen3 4B	64.531	128	
Qwen3 1.7B	37.266	128	
Gemma-4 E4B	36.901	128	
Gemma-4 E2B	28.789	128	
Gemma-3 27B	22.734	128	
Gemma-3 12B	17.161	128	
Qwen2.5 32B	13.281	128	
Qwen3 600M	12.917	128	
Qwen2.5 14B	11.471	128	
Gemma-3 4B	10.534	128	
Qwen2.5 7B	7.669	128	
Qwen2.5 3B	3.255	128	
Llama-3.1 8B	0.812	4,096	✓
Qwen2.5 1.5B	0.665	4,096	✓
Gemma-3 1B	0.484	4,096	✓
Gemma-1.1 7B	0.247	4,096	✓
Llama-3.2 1B	0.222	4,096	
Gemma-2 9B	0.199	4,096	
Qwen2.5 500M	0.175	4,096	✓
Mistral 7B	0.110	4,096	✓
Qwen1.5 1.8B	0.107	4,096	
Gemma-3 270M	0.103	4,096	
Gemma-1.1 2B	0.101	4,096	
Llama-2 7B	0.089	4,096	
Gemma-2 2B	0.076	4,096	
Llama-2 13B	0.059	4,096	✓

HLE

Model	Acc. (%)	Rollouts	Set
Gemma-4 31B	18.700	192	
Gemma-4 26B-A4B	17.324	192	
Qwen3 32B	10.612	192	
Qwen3 30B-A3B	6.274	192	
Qwen3 14B	4.273	192	
Gemma-4 E4B	3.962	192	
Gemma-3 4B	3.616	192	
Qwen3 4B	3.527	192	
Qwen2.5 3B	3.220	192	
Gemma-4 E2B	3.006	1,024	✓
Qwen2.5 14B	2.957	1,024	
Qwen2.5 7B	2.898	1,024	
Gemma-3 27B	2.828	1,024	✓
Qwen3 8B	2.772	1,024	
Llama-2 13B	2.764	1,024	
Qwen2.5 1.5B	2.624	1,024	✓
Gemma-3 12B	2.619	1,024	
Qwen3 600M	2.575	1,024	
Gemma-3 1B	2.521	1,024	
Qwen2.5 32B	2.431	1,024	✓
Llama-3.1 8B	2.241	1,024	✓
Qwen3 1.7B	2.220	1,024	
Qwen2.5 500M	2.124	1,024	
Gemma-3 270M	2.001	1,024	✓
Llama-2 7B	1.658	1,024	✓

95% confidence interval on the accuracy difference between two models does not include zero. This is desirable because if the confidence interval on the accuracy difference between two models does not include zero, their ranking cannot flip. Because the intervals are simultaneous over all pairs, the Bonferroni-corrected guarantee holds for any subset of models chosen afterwards, so no further correction is needed for the choice of a model set from this model pool.

From the certified model pairs we take the longest chain of models in which every consecutive pair is certified and keep its seven lowest-accuracy members. The chosen sets are marked in Table 2. These model sets are both near-zero and nearly indistinguishable: their unhinted accuracies are at most 0.8% on AIME and 3.0% on HLE, making them near-zero. They span 0.06% to 0.81% on AIME and 1.7% to 3.0% on HLE, less than 1% and 2% respectively, making them nearly indistinguishable.

G.2 The hint-prior ranking

Given a budget of k rollouts per problem, the hint-prior ranking spends one rollout per problem on the hinted benchmarks and the remaining $k-1$ at $H=0$. Each problem's hinted rollout uses a single hint level, assigned so that every hint level covers an equal share of the problems. Averaging the hinted rollouts gives the hinted score $\widehat{\mathrm{Acc}}_1(M,\bar{H})$, where $\bar{H}$ stands for the pool of all hint levels; averaging the unhinted rollouts gives the unhinted score $\widehat{\mathrm{Acc}}_{k-1}(M,0)$, as in avg@$k$. Both scores are standardized across models before they are combined in Eq. (2).

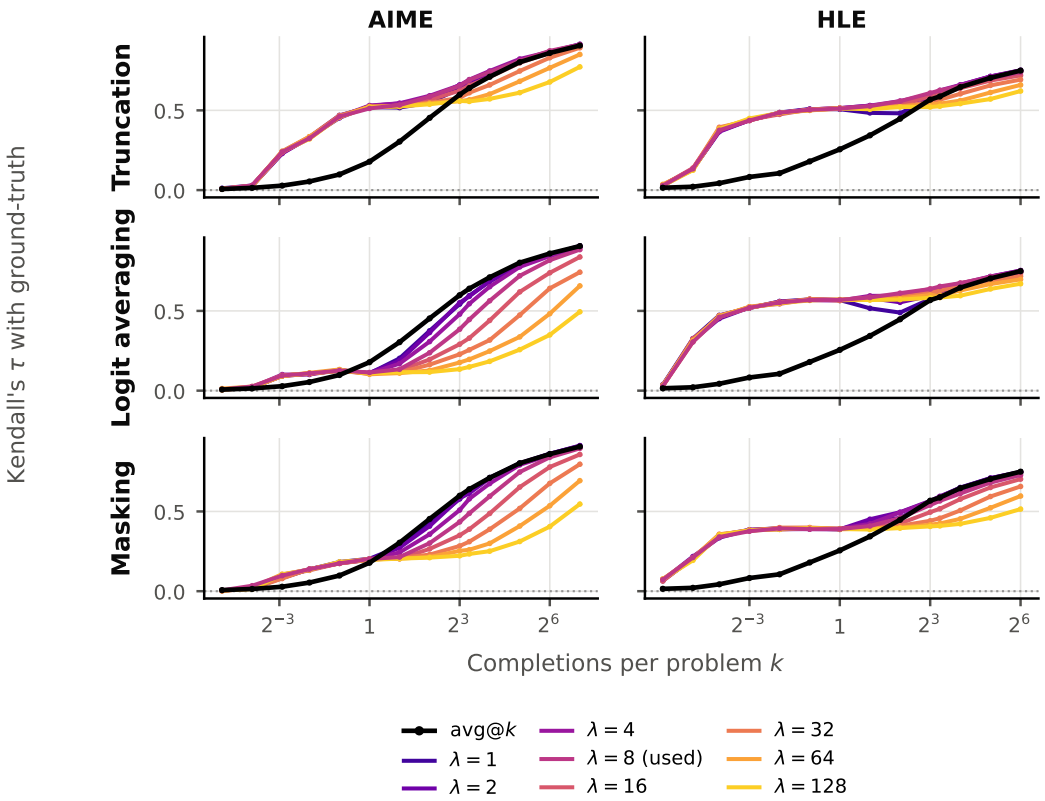

Figure 17: **Sensitivity of the hint-prior ranking to λ.** Kendall's τ against the budget k for the hint-prior ranking (Eq. (2)) with $\lambda \in \{1, 2, 4, 8, 16, 32, 64, 128\}$, for all three hinting methods (rows) on both benchmarks (columns), with avg@k in black. λ is the number of unhinted rollouts per problem the hinted score is worth: small λ hands over to the unhinted score early, large λ late.

The weighting reflects the two scores' errors. The unhinted score is unbiased but noisy, and its noise shrinks with k. The hinted score is far less noisy but biased if hinted rankings do not match the ground-truth unhinted ranking. This bias does not shrink with k. Therefore, the weight α_k moves from the hinted score to the unhinted one as the budget grows. Because α_k depends only on the budget and not on the data, the hint-prior ranking becomes avg@k at large budgets even when the hinted ranking is wrong, which happens for some hinting methods, as §G.3 shows. Below one rollout per problem there is no unhinted score, and the single hinted rollout covers a random subset of the problems.

Choice of λ. The weight $\alpha_k = (k-1)/(k-1+\lambda)$ in Eq. (2) treats the hinted score as a prior worth λ unhinted rollouts per problem. Fig. 17 sweeps λ from 1 to 128. No single choice of λ achieves optimal Kendall's τ. Small λ values shift weight to the unhinted score quickly, which is suboptimal if the hinted scores are correct, but big λ values keep weight on the hinted scores for too long, which is suboptimal if the hinted scores are incorrect. We use $\lambda = 8$ throughout as an intermediate value.

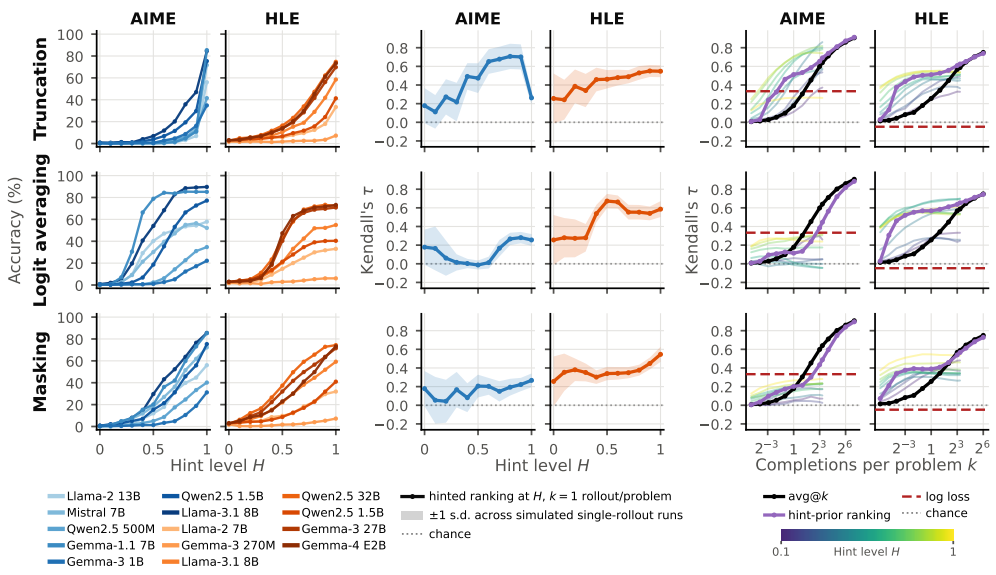

Figure 18: **Fig. 7 for all three hinting methods.** Rows are hinting methods, columns benchmarks; left, middle and right panels as in Fig. 7.

G.3 Ranking with other hinting methods

Fig. 18 repeats Fig. 7 for logit averaging and masking. Hinted rankings largely agree with the ground-truth unhinted ranking under every method: in four of the six method–benchmark pairs, a single hinted rollout per problem ranks the models closer to the ground truth than a single unhinted one, and often by a wide margin (middle panels). Truncation is the most reliable method, with a median τ over hint levels of 0.48 on AIME and 0.47 on HLE. Logit averaging and masking reach 0.55 and 0.35 on HLE but are no better than an unhinted rollout on AIME, where families gain from these hints at different rates and the curves cross. The best hint level also differs between methods: with logit averaging the best level on HLE is $H = 0.5$.

The hint-prior ranking follows this agreement (right panels). On HLE, where the hinted rankings of logit averaging and masking are informative, it leads avg@k at small budgets, by 0.31 with logit averaging and 0.13 with masking at one rollout per problem. On AIME, where they are not, it trails avg@k at intermediate budgets before converging with it. Wherever hinted rankings agree with unhinted ones, the hint-prior ranking recovers the ranking from a fraction of the rollouts avg@k needs, and where they do not, it falls back to avg@k as the budget grows.