WHEN ARE LIKELY ANSWERS RIGHT? ON SEQUENCE PROBABILITY AND CORRECTNESS IN LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Many decoding methods for large language models can be understood as shifting probability mass toward outputs that are more likely under the model, either locally at the token level or globally at the sequence level. Therefore, their success depends on a fundamental question: when does sequence probability, that is, the conditional probability of a continuation given a prompt, actually align with correctness? In this paper, we set out to quantify this relationship across decoding methods, models, and benchmarks at four levels: across decoding methods, across hyperparameters within a method, across prompt-answer pairs within a dataset, and across repeated responses to the same prompt. We find that higher sequence probability is often predictive of correctness across prompt-answer pairs within a fixed dataset. However, this relationship does not generally transfer to decoding decisions: increasing sequence probability by changing hyperparameters or methods does not reliably improve accuracy. Further, sequence probability is not a good indicator of correctness for responses to the same prompt. These findings clarify when decoding can and cannot be expected to improve correctness, and provide practical guidance for decoding, self-consistency, and verifier-free self-improvement.

1 INTRODUCTION

Large language models (LLMs) are autoregressive models that assign probabilities to sequences of tokens. At inference time, decoding methods use the autoregressive next-token distribution to exploit this distribution in different ways. Many widely used methods can be viewed as shifting probability mass toward outputs that are more likely under the model. Decoding methods, like low-temperature next-token sampling and truncation methods (e.g., top-k sampling (Fan et al., 2018)) suppress unlikely tokens *locally*. Best-of-N (BoN) keeps only the highest-probability samples among a set of N sequences approximating the *global* mode (for $N \to \infty$) similar to beam search (Lowerre, 1976; Sutskever et al., 2014). Power sampling (Karan & Du, 2025; Ji et al., 2026; Azizi et al., 2026) biases the distribution over full sequences towards high-probability regions over-weighting high-probability sequences and under-weighting low-probability sequences.

Viewing decoding through this lens raises a fundamental question: *when does sequence probability align with correctness?* An answer to this question has two important implications. Firstly, it helps explain when particular decoding methods improve performance and opens up a way to guide the development of new decoding methods. Secondly, if higher-probability sequences are more likely to be correct, then model probability itself may provide a *verifier-free approach to self-improvement* by using higher-probability samples for self-consistency (Wang et al., 2023; 2024a; Taubenfeld et al., 2025) and self-distillation (Hinton et al., 2015; Zhang et al., 2019).

Since LLMs are trained on data which is predominantly correct, it seems natural to expect correct continuations to receive higher probability than incorrect ones. However, this intuition is too simple: it has been observed in various contexts that beam search, which approximately finds the most likely sequence, often degrades performance (Vijayakumar et al., 2016; Cohen & Beck, 2019; Koehn & Knowles, 2017; Yang et al., 2018; Su et al., 2022). This suggests that the question is less *whether* there is a general probability-correctness relationship, but *under which conditions* probability and correctness align and which decoding methods are able to exploit the relationship.

In this paper, we quantify the relationship between sequence probability and correctness across 8 decoding methods (2 methods targeting the power distribution, 2 methods targeting the mode of

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.

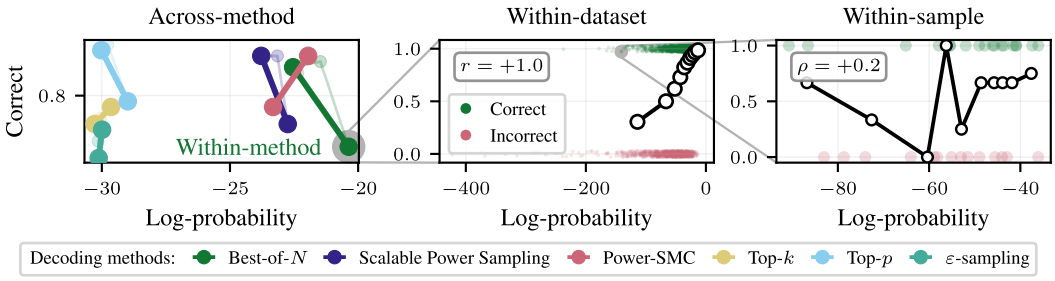


Figure 1: **The correlation between log-probability and correctness across three levels.** Left: *Across-method-correlation*: log-probability plotted against accuracy for various decoding methods and hyperparameters (connected by lines). *Within-method correlation*: plotted without transparency. Middle: *Within-dataset correlation*: for one method (BoN with $N = 32$) we bin correct (green dots, top) and incorrect data (red dots, bottom) in 10 bins and fit a Pearson correlation. The correlation is positive. Right: *Within-sample correlation*: for one sample of the dataset, a Pearson rank correlation is computed (also, binned Pearson corelation is shown). The correlation is positive. The plot shows data of `Qwen3-8B` on MATH500. See Section 1 and Section 3 for details.

the distribution, and 4 local methods), 14 models (from the `Qwen2.5`, `Qwen3`, `Olmo3` families), and 6 benchmark datasets. We analyze the relationship on three different levels: We define *across-method correlation* which fixes model and dataset and computes correlation across methods with a representative hyperparameter (Figure 1, left). For each method, *within-method correlation* considers the correlation across hyperparameters (Figure 1, left, non-transparent lines). Zooming in on one hyperparameter (Figure 1, middle) we define the *within-dataset correlation*. Finally, *within-sample correlation* is the most granular correlation for an individual sample from a dataset (Figure 1, right).

Our results show that there is no uniform probability-correctness relationship for LLM generations. We find a consistent correlation within a dataset depending on the model family but not the method (Section 3.1). This indicates that a model is often able to tell correct prompt-answers pairs from incorrect prompt-answer pairs. However, this signal does not generally transfer to the settings directly used for decoding. We find that tuning the hyperparameters of a decoding method, while producing sequences of higher log-probability, does not result in more correct sequences (Section 3.2). Further, methods that produce higher-probability sequences are not consistently more accurate (Section 3.3). For a single prompt, there is no consistent correlation within the corresponding responses. However, more correct samples also show larger within-sample correlations (Section 3.4).

Summarily, the main contributions of this work are as follows:

- We frame decoding methods as approaches to maximize sequence probability: we distinguish local methods from global sequence-level methods and give a variational objective for global methods.
- We study the relationship between sequence probability and correctness across 12 models, 6 datasets, and 8 decoding methods. We analyze correlation on various levels: *across-method correlation*, *within-dataset correlation*, *within-method correlation*, and *within-sample correlation*.
- We find that within-dataset correlations between log-probability and correctness are often consistent and largely determined by the dataset and model variant, rather than by the decoding method. We show that increasing sequence probability by changing a decoding method or tuning its hyperparameters does not reliably improve correctness.
- We connect these findings to verifier-free self-improvement methods. We find that probability can be informative when the model already has sufficient task accuracy, but within-sample correlations are often weak or symmetric around zero, limiting the reliability of probability-weighted aggregation.

2 Decoding Methods Maximize Sequence Probability in LLMs

In this section, we view decoding methods in large language models (LLMs) as sampling mehods to produce high-probability sequences. We highlight two classes of commonly used decoding methods: local decoding methods in Section 2.1 and global decoding methods in Section 2.2.

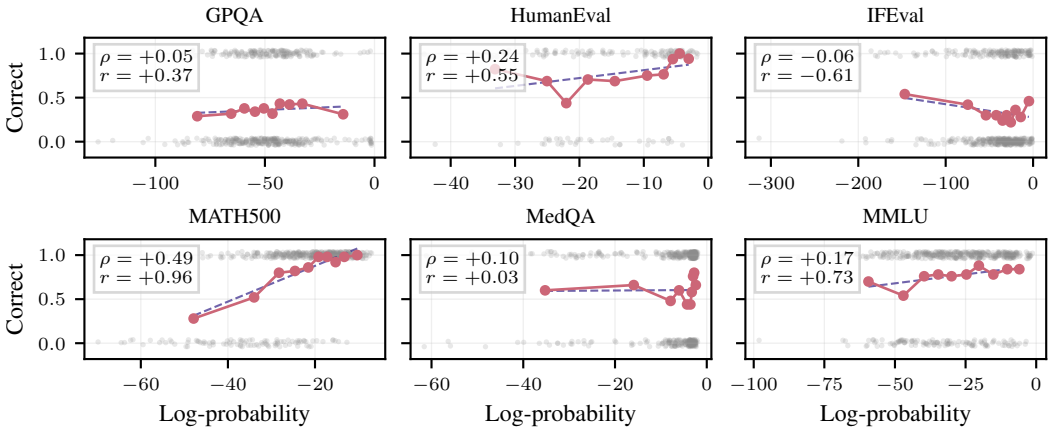


Figure 2: **Within-dataset correlation: log-probability and correctness are correlated for each dataset.** Each panel scatters samples using log-probability and correctness. ρ: Spearman correlation coefficient. r: (binned) Pearson correlation coefficient. Correlation is strongest for MATH500 and smaller but positive for GPQA, HumanEval, MedQA, and MMLU. Only IFEval shows a negative correlation. Plots show `Qwen3-8B-Base` and SPS with $B = 192$. See Section 3.1 for details.

Let p denote an autoregressive LLM. Further, let the distribution over continuations $\mathbf{s} = (s_1, \ldots, s_T)$ given a prompt $\bar{\mathbf{s}}$ under p be defined as follows

$$p(\mathbf{s} \mid \bar{\mathbf{s}}) = \prod_{t=1}^{T} p(s_t \mid \bar{\mathbf{s}}, \mathbf{s}_{<t}). \tag{1}$$

2.1 Local Decoding Methods

Local sampling methods change the next-token distribution at every prefix $\bar{\mathbf{s}} \mid \mathbf{s}_{<t}$ for $t = 2, \ldots, T$ and, thus, generally do *not* sample the most probable *global* sequences (see Remark B.1).

Low-temperature next-token sampling (LTS). Standard sampling at low temperatures modifies the next-token distribution via

$$\tilde{p}_\alpha(s_t \mid \bar{\mathbf{s}}, \mathbf{s}_{<t}) := \frac{p(s_t \mid \bar{\mathbf{s}}, \mathbf{s}_{<t})^\alpha}{\tilde{Z}_\alpha(\bar{\mathbf{s}}, \mathbf{s}_{<t})} \quad \text{with} \quad \tilde{Z}_\alpha(\bar{\mathbf{s}}, \mathbf{s}_{<t}) := \sum_{s \in \mathcal{V}} p(s \mid \bar{\mathbf{s}}, \mathbf{s}_{<t})^\alpha. \tag{2}$$

Since LTS has the same support as p, low-probability regions are down-weighted but not removed.

Truncated next-token sampling (TS). TS specifies an active set $A(\bar{\mathbf{s}}, \mathbf{s}_{<t}) \subseteq \mathcal{V}$ at each prefix as

$$p_A(s_t \mid \bar{\mathbf{s}}, \mathbf{s}_{<t}) := \frac{p(s_t \mid \bar{\mathbf{s}}, \mathbf{s}_{<t})}{\tilde{Z}_A(\bar{\mathbf{s}}, \mathbf{s}_{<t})} \mathbb{1}[s_t \in A(\bar{\mathbf{s}}, \mathbf{s}_{<t})] \quad \text{with} \quad \tilde{Z}_A(\bar{\mathbf{s}}, \mathbf{s}_{<t}) := \sum_{s \in A(\bar{\mathbf{s}}, \mathbf{s}_{<t})} p(s \mid \bar{\mathbf{s}}, \mathbf{s}_{<t}). \tag{3}$$

At every step, TS only samples from within this active set. The support of p_A is different from p.

Top-k sampling (Fan et al., 2018) defines an active set given by the k highest-probability tokens under $p(\cdot \mid \bar{\mathbf{s}}, \mathbf{s}_{<t})$ as $A(\bar{\mathbf{s}}, \mathbf{s}_{<t}) = \{\text{top-}k \text{ tokens under } p(\cdot \mid \bar{\mathbf{s}}, \mathbf{s}_{<t})\}$. **Top-$p$ sampling** (Holtzman et al., 2020) chooses the smallest active set $A(\bar{\mathbf{s}}, \mathbf{s}_{<t})$ whose cumulative mass under $p(\cdot \mid \bar{\mathbf{s}}, \mathbf{s}_{<t})$ exceeds a threshold $p \in (0, 1]$. **ε-sampling** (Hewitt et al., 2022) sets $A(\bar{\mathbf{s}}, \mathbf{s}_{<t}) = \{s \in \mathcal{V} : p(s \mid \bar{\mathbf{s}}, \mathbf{s}_{<t}) > \varepsilon\}$.

2.2 Global Decoding Methods

Global sampling methods, as discussed in the following, target the globally most likely sequences.

Maximizing sequence probability. Beam search (Lowerre, 1976; Sutskever et al., 2014) (BS) is an approximate search procedure for $\mathbf{s} = \arg\max_{\mathbf{s}'} p(\mathbf{s}' \mid \bar{\mathbf{s}})$ by controlling its range with the so-called number of beams K. Greedy decoding is recovered for $K = 1$. BS is known to often find repetitive sequences (Vijayakumar et al., 2016; Cohen & Beck, 2019). Further, quality deteriorates with larger beams (Koehn & Knowles, 2017; Yang et al., 2018) and sequences lack diversity (Su et al., 2022).

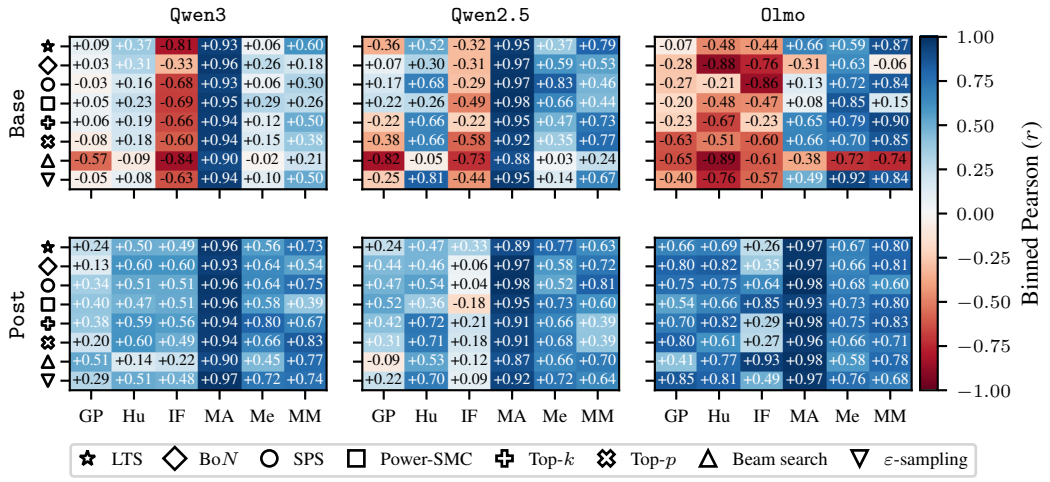


Figure 3: **Within-dataset correlation: log-probability and correctness is consistently correlated across model families and datasets, largely independent of methods.** `Base` models show more negative and diverse correlation and are consistently negative for IFEval. `Posttrained` models show consistently positive correlation. Correlation coefficients r are averaged over model sizes at a canonical hyperparameter for each method. Methods are in the legend. Datasets are plotted on the x-axis of each panel: GPQA, Humaneval, IFEval, MATH500, MedQA, MMLU. See Section 3.1.

Remark 2.1 (Beam search generates the highest probability sequence for $K \to \infty$.)**.** *For large K beam search returns an element of* $\arg\max_{\mathbf{s}} p(\mathbf{s} \mid \bar{\mathbf{s}})$.[1]

Best-of-N (BoN) sampling. BoN samples N sequences and returns the sequence with largest probability. For $N \to \infty$ one recovers the mode of the distribution when sampling from p (Equation (1)).

Power distribution over sequences. The power distribution with $\alpha > 1$ is defined over full sequences

$$p_\alpha(\mathbf{s} \mid \bar{\mathbf{s}}) := \frac{p(\mathbf{s} \mid \bar{\mathbf{s}})^\alpha}{Z_\alpha(\bar{\mathbf{s}})} \quad \text{with} \quad Z_\alpha(\bar{\mathbf{s}}) := \sum_{\mathbf{s}} p(\mathbf{s} \mid \bar{\mathbf{s}})^\alpha. \tag{4}$$

Contrary to BoN, power-sampling increases sequence probabilities *in expectation*.

Remark 2.2 (Power sampling increases expected sequence log-probability)**.** *Let* $\mathbf{s} \sim p_\alpha(\cdot \mid \bar{\mathbf{s}})$ *($\alpha > 1$. Increasing α shifts mass toward sequences with larger probability under p.*

While power sampling looks similar to LTS at first sight, the methods are generally not equal.

Remark 2.3 (LTS is generally not equal to sequence-level power sampling)**.** *Unless the local normalization constants of LTS are identical across all reachable prefixes, no single token-level temperature reproduces the sequence-level power distribution.*

Sampling from the power distribution. Karan & Du (2025) are the first to sample from the power distribution of LLMs. In the following investigation, we focus on scalable power sampling (SPS) (Ji et al., 2026) and power-SMC (Azizi et al., 2026). SPS samples blocks of B tokens and uses rollouts after each block to estimate the future probability mass behind the top-k tokens, power-SMC utilizes a sequential Monte Carlo sampler generating N particles. See Appendix C for a general framework.

Variational objective for global decoding. From a variational perspective, the power distribution p_α can be understood as the optimizer of a variational objective trading off expected sequence probability and entropy by α. For any $q(\mathbf{s} \mid \bar{\mathbf{s}})$ let $\mathcal{J}_\alpha(q) := \mathbb{E}_q[\log p(\mathbf{s} \mid \bar{\mathbf{s}})] + 1/\alpha\, \mathcal{H}(q)$, where $\mathcal{H}(q) = -\sum_{\mathbf{s}} q(\mathbf{s} \mid \bar{\mathbf{s}}) \log q(\mathbf{s} \mid \bar{\mathbf{s}})$ is the entropy of q. We find the following.

Remark 2.4 (The power distribution is a maximizer of the variational objective.)**.** *The maximizer of $\mathcal{J}_\alpha(q)$ is the power distribution, $q^\star_\alpha \in \arg\max_q \mathcal{J}_\alpha(q) \iff q^\star_\alpha(\mathbf{s} \mid \bar{\mathbf{s}}) = p_\alpha(\mathbf{s} \mid \bar{\mathbf{s}})$.*

Thus, the power distribution can be seen as a *soft* alternative to mode-seeking via its parameter α. In the limit $\alpha \to \infty$, the entropy term vanishes and the optimizer collapses onto the mode of $p(\mathbf{s} \mid \bar{\mathbf{s}})$. As

[1]We use the implementation of beam search in vLLM (Kwon et al., 2023) which adds a length normalization.

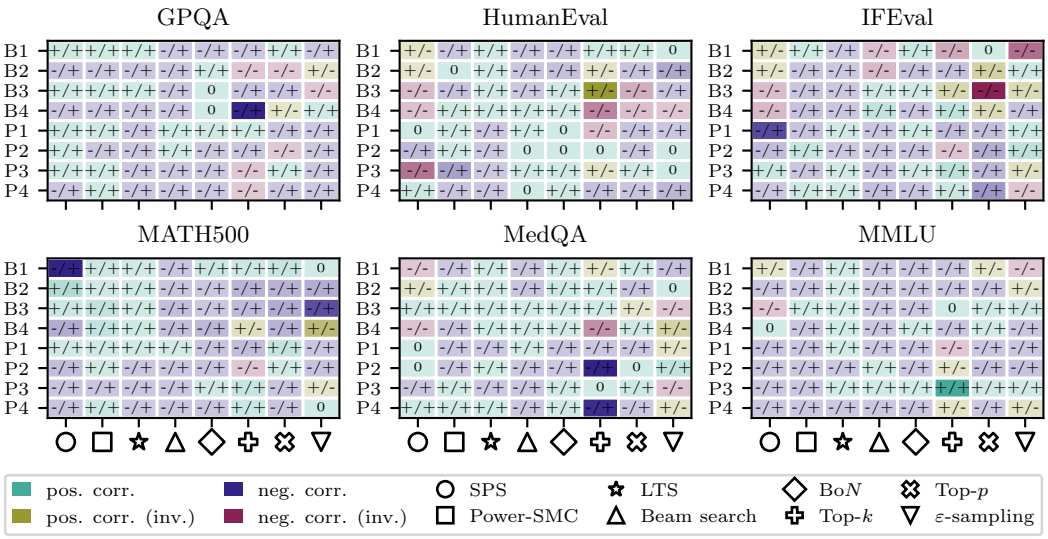


Figure 4: **Within-method correlation: while sequences are achieving higher log-probability, one cannot predict correctness by changing a hyperparameter of the method.** Correlations within methods for local and global decoding methods, models (`Qwen3` series 0.6B as $*1$, 1.7B as $*2$, 4B as $*3$, and 8B as $*4$ with base as B$*$ and posttrained model as P$*$), and benchmark datasets. Correlations are not consistent across datasets and models. Color coding in Figure 6. Discussion in Section 3.2.

N grows, BoN increasingly favors high-probability regions and, in the limit, it returns the $\arg\max$. For a function of a continuation $\mathbf{s}$, we find that $\partial/\partial\alpha\, \mathbb{E}_{p_\alpha}[f(\mathbf{s})] = \mathrm{Cov}_{p_\alpha}(f(\mathbf{s}), \log p(\mathbf{s} \mid \bar{\mathbf{s}}))$., i.e., features $f(\mathbf{s})$ that are positively correlated with sequence log-probability increase under sharpening, while features negatively correlated with it decrease.

3 Probability and Correctness at Various Granularity

In this section, we provide our empirical study on relationship between sequence probability and correctness focusing on the decoding methods in Section 2 at various levels visualized in Figure 1.

Models and datasets. We work with three model families: we use the `Qwen3` model series (Yang et al., 2025) with model sizes `0.6B`, `1.7B`, `4B`, and `8B` spanning `base` and `posttrained` models. Additionally, we provide experiments for the `Qwen2.5(-Math)-8B` models (`base` and `posttrained`) and we consider `Olmo3-7B` (`base` and `posttrained`) models. Due to the computational requirements of SPS and power-SMC we limit trajectory lengths to 3072 tokens following previous work (Karan & Du, 2025; Ji et al., 2026; Azizi et al., 2026) and evaluate non-thinking `posttrained` `Qwen3` models to make sure that the model emits a response. Appendix E.9 shows an ablation study using the `Qwen3` thinking models. We report results on 6 benchmark dataset: We use MATH500 (Hendrycks et al., 2021), GPQA (Rein et al., 2024), and MMLU (Wang et al., 2024c) for mathematical reasoning, Humaneval (Chen et al., 2025) for code generation, MedQA (Jin et al., 2021) for commonsense and out-of-distribution reasoning, and IFEval (Zhou et al., 2023) for instruction following.

Decoding methods. We consider the methods presented earlier in Section 2. We run SPS with block sizes $B \in \{96, 192, 384\}$. Power-SMC is plotted with $N \in \{8, 16, 32\}$. We run LTS with temperatures $\tau \in \{0.25, 0.7\}$. Beam search is plotted with number of beams $b \in \{1, 2, 3\}$. BoN is run for $N \in \{8, 16, 32\}$. Top-k is plotted with $k \in \{8, 16, 32\}$. Top-p is run with $p \in \{0.8, 0.9, 0.95\}$. ε-sampling is plotted with $\varepsilon \in \{3 \cdot 10^{-4}, 9 \cdot 10^{-4}, 5 \cdot 10^{-2}\}$. More details in Appendix D.1.

3.1 Within-Dataset Correlation

In this section, we investigate the *relationship between log-probability and correctness within a dataset*, i.e., for a given model, dataset, and method. Exemplary, in Figure 2 we show correlations

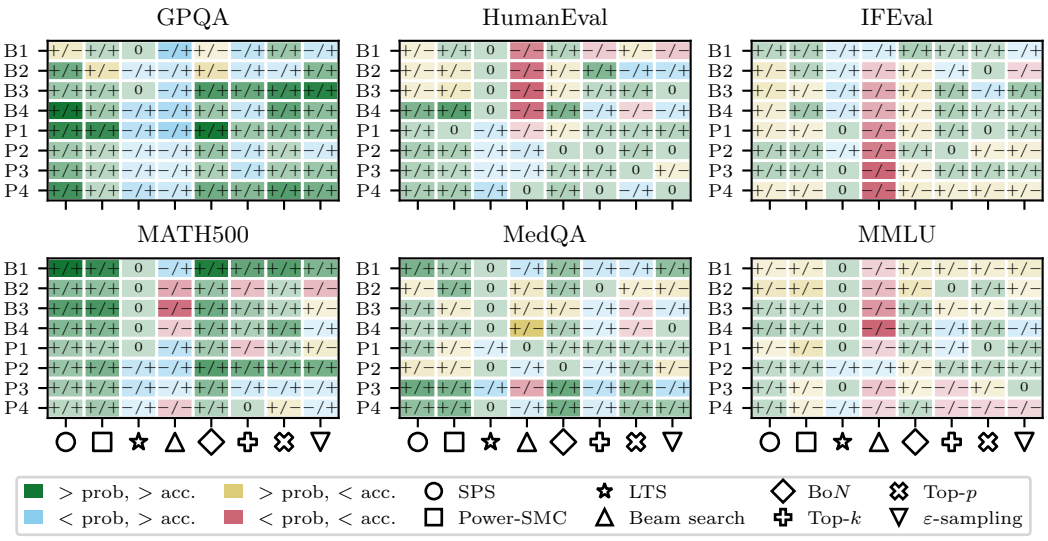


Figure 5: **Across-method correlation: many methods produce samples that are both more probable and more accurate than the low-temperature sampling (LTS) baseline. However, correlations are not consistent across models and datasets.** Correlations across methods for local and global decoding methods, models (`Qwen3` series 0.6B as $*1$, 1.7B as $*2$, 4B as $*3$, and 8B as $*4$ with base as B$*$ and posttrained model as P$*$), and benchmark datasets. The color coding is provided in Figure 8. A detailed discussion can be found in Section 3.3.

for SPS with block size $B = 192$, `Qwen3-8B-Base`, on all 6 benchmarks. We scatter the log-probability of all samples against their correctness label $\in \{0, 1\}$ (gray), for each dataset. Then, we bin samples into 10 bins of equal size, computing log-probability and accuracy by averaging over the points falling within a bin and compute a binned Pearson correlation coefficient r, see Appendix E.1.

We consistently find correlation between log-probability and correctness with strength and direction depending on the dataset. Strikingly, we see a strong, near-perfect, and positive correlation on MATH500 (Figure 2). For GPQA, HumanEval, MedQA, and MMLU we find smaller but similarly positive correlations. Thus, the log-probability of prompt-answer pairs if often informative of correctness. IFEval is the only dataset with negative correlation. This could indicate that `Qwen3-8B-Base` has seen only limited instruction-following data during training, or, that the base models struggle to separate likely completions from likely completions that violate the prepended formatting instructions.

Next, we summarize the results across models and methods. We choose a canonical hyperparameter per method (see Appendix D.3) and average results across model sizes. In Figure 3 we show the resulting correlation coefficient across model families over all datasets considered: GPQA, Humaneval, IFEval, MATH500, MedQA, MMLU. We observe a *consistent* structure in the results depending on dataset and model-family, *largely independent of the method*. Specifically, we find that for MATH500 correlation appears to be strongest across model-family and method, independent of the model variant. Further, posttrained models show predominantly positive correlation coefficients with minor exceptions. Differences between base and posttrained models are visible mostly on IFEval where base models show (often large) negative correlations while posttrained models show small but positive correlations. An exception is mathematical reasoning MATH500 where correlations are large and positive. One notable outlier in the method-rows appears to be beam search which includes a length normalization. We summarize the key takeaway in Finding 1.

Finding 1: The within-dataset correlation is largely independent of the method and mostly dependent on dataset and model variant. Posttrained models show mostly positive correlation, while base models are more diverse. Thus, log-probability is a meaningful indicator for correctness when judging multiple samples from the same dataset, even without knowing the decoding method.

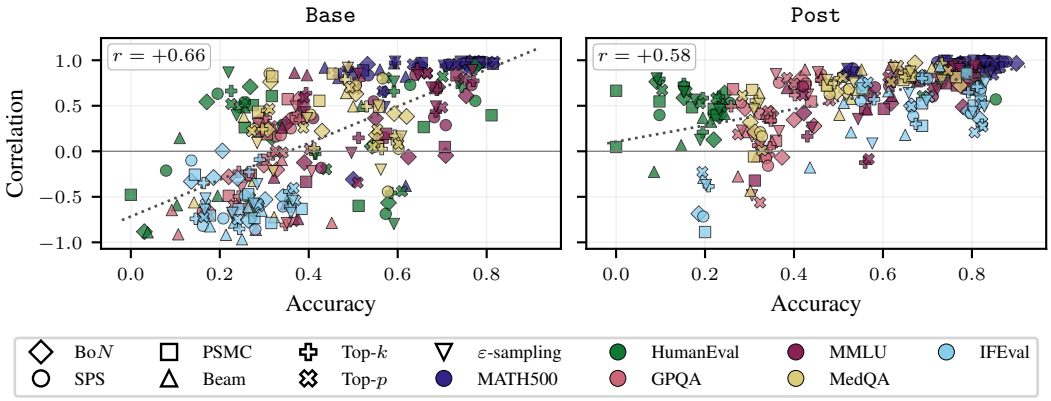


Figure 7: **Within-dataset correlation: the correlation between log-probability and correctness increases with accuracy. Probability-based self-improvement loops would require the model to have sufficient accuracy.** Accuracy is plotted against correlation across all datasets, methods, and models (left: `base`, right: `posttrained`, at canonical hyperparameter). We observe positive correlation (`base`: $r = 0.66$, `posttrained`: $r = 0.59$), `base` models show negative correlation coefficients, `posttrained` models show mostly positive correlation coefficients. See Section 3.3.

3.2 Within-Method Correlation

Zooming out to a method level, here we ask *whether changing a hyperparameter for a particular method moves both log-probability and correctness*. This corresponds to the left panel in Figure 1 where we show within-method correlation as a solid line (other hyperparameters transparent).

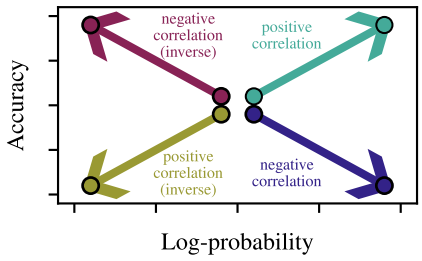


Figure 6: **Color coding for Figure 4.** Further details in Section 3.2.

For each of the decoding methods, we fit a line through the extreme hyperparameter settings and summarize its slope together with the sign of the corresponding correctness change. Figure 6 shows the color-coding. We plot positive correlation with positive slope, positive correlation with inverse slope (i.e., upper-right to lower-left), negative correlation with negative slope, and negative correlation with inverse slope for each method for the `Qwen3` series in Figure 4. For other models see Appendix E.3.

The results show either positive (green, $+/+$) or negative (blue, $-/+$) correlation with approximately same numbers. Therefore, changing the hyperparameter of a decoding method almost always leads to more probable sequences. At the same time, this does not result in more correct sequences. Investigating methods (columns) across datasets we are unable to find any consistent results that hold across models. Comparing global and local sampling methods, it seems like local methods show inverse relationships more often. This can be explained by the fact that these methods have *local* normalization constants. Therefore, there is no *consistent*, i.e., independent (of dataset and model) way to produce both *more probably and more correct* sequences *within a method* by changing the corresponding hyperparameter. We summarize the main takeaways from this section in Finding 2.

Finding 2: Within a decoding method, tuning its hyperparameter results in more probable sequences. However, this often result in less correct sequences. Practically speaking, the results suggest that there is no consistent benefit of tuning the hyperparameters of a method. Instead, hyperparameters need to be tuned with respect to method, model, and dataset.

3.3 Correlation Across Methods

In this section, we consider the *correlation across methods*, see Figure 1 (left). To simplify the analysis, we choose low-temperature next-token sampling (LTS) with $\alpha = 4$ as zero point and compute in which of the four quadrants the *best-performing hyperparameter* of each method falls

into. Figure 8 shows the color coding: We plot higher probability sequences which are more correct, lower probability sequences that are more correct, higher probability sequences that are less correct, and lower probability sequences which are less correct. Note that LTS acts as meaningful zero-point as it is part of (almost) all sampling methods choosing a specific set of hyperparameters (except for beam search). Other models in Appendix E.3.

Figure 5 shows the aggregate plot using the colors discussed above over all methods and datasets. Opacity is set according to |accuracy|. Firstly, we find that many methods produce samples that are more probable while also being more correct than the LTS baseline with all cells of green (+/+) providing positive evidence. Together with the red cells (−/−) they provide evidence for a positive correlation (including LTS). Generally, however, the correlation is heterogeneous. It is strongest on MATH500. For MMLU, HumanEval, MedQA, GPQA, and IFEval the relationship seems weaker and methods show more yellow (+/−) and blue (−/+) cells which are evidence against a probability-correctness relationship (including LTS as method paramter). Beam search consistently results in less probable sequences that are often less correct which might be due to its length-normalization. The main finding we would like to highlight is that no method (even if an optimal hyperparameter has been chosen by an oracle) reliably beats the LTS baseline across datasets. Similarly, achieving a larger log-probability (than LTS, across methods) does not reliably translate to more correct sequences. Many decoding methods find large probability sequences which are, however, less correct (yellow, +/−) compared to LTS samples. Finding 3 summarizes the takeaways.

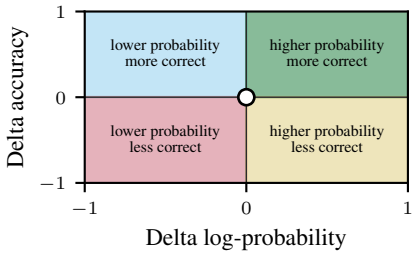


Figure 8: **Color coding for Figure 5.** Details in Section 3.3.

Finding 3: Higher log-probability across methods does not reliably translate to more accurate solutions. Empirically, we find it impossible to choose a method that consistency outperforms low-temperature next-token sampling. Power-sampling methods reliably find more probable sequences which, however, are not always more correct. Local methods often produce less probable sequences which are more correct.

Correlation approximately scales with accuracy. In Figure 7 we plot accuracy against within-dataset correlation across all datasets and methods considered. We observe a positive trend (a correlation coefficient of $r = 0.66$ for `base` models, and $r = 0.59$ for `posttrained` models). Thus, a larger mean accuracy within the dataset correlates with a stronger relationship between log-probability and correctness when measured across methods and models. We find that datasets cluster (colors) while methods (markers) do not seem to follow a specific structure. The difference between `base` models and `posttrained` models is surprising to some degree as `base` models are typically thought to be more calibrated than `posttrained` models which does not seem to hold true here. On a different note, one can pose that for probability-based verifier-free recursive self-improvement loops a sufficient baseline accuracy for the tasks at hand might be required for enabling improvement. Else, one would reinforce on samples that are increasingly less correct. Finding 4 summarizes takeaways.

Finding 4: The within-dataset correlation between log-probability and correctness approximately increases with accuracy. Therefore, verifier-free recursive self-improvement loops relying on probability, might need sufficient base accuracy to improve. This applies to `base` models to a larger degree as it applies to `posttrained` models which mostly show positive correlations.

3.4 Within-Sample Correlation

In this section, we focus on *within-sample correlation* at the finest level. We only focus on low-temperature next-token sampling (LTS) with $\alpha = 4.0$ by reusing the BoN samples.

In Figure 9 we show boxplots over the per-sample rank correlation coefficients ρ (see Appendix E.1) of `base` (left) and `posttrained` (middle) models, respectively, considering all datasets. We find that corelation coefficients seem to be symmetrically distributed around zero with zero mean while

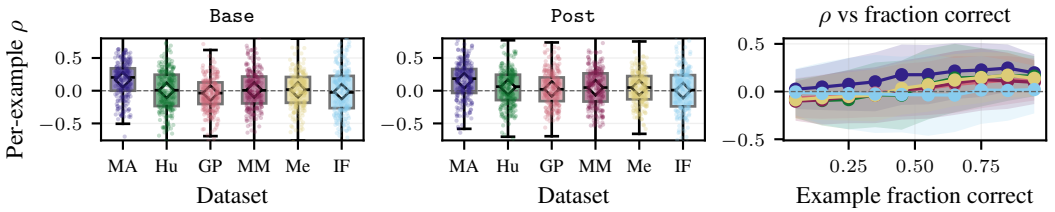


Figure 9: **Within-sample correlation: correlation coefficients are distributed symmetrically around zero (left, middle). Further (right), the more correct a set of continuations for a single prompt is, the more positive is its correlation.** Per-sample rank correlation coefficient of `Qwen3 base` models (left) and `posttrained` models (middle) is distributed symmetrically with mean zero. Right: Correlation coefficient and fraction of correct samples is positively correlated. Datasets are plotted on the x-axis: GPQA, Humaneval, IFEval, MATH500, MedQA, MMLU. See Section 3.4.

only MATH500 shows a positive mean. When analyzing per-sample correlation with respect to the fraction of correct responses (out of the 32 LTS $\alpha = 4.0$ examples), Figure 9 (right) shows a positive trend (with large and overlapping standard deviations).

Therefore, while there is little consistent structure in the with-sample correlations, we find that the more correct the repeated sampling for each prompt is, the more correlation can be observed within the log-probabilities. This finding is similar to the result in Figure 7 where correlation approximately scales with accuracy. We summarize this finding in Finding 5. We provide more granular results and a study on within-sample correlations *across methods* in Appendix E.4.

Finding 5: Within-sample correlations are distributed symmetrically around zero with zero mean. MATH500 is the only dataset with correlations averaging to a positive value. Empirically, there appears to be a positive relation considering the within-sample correlation with respect to the correctness (i.e., number of correct responses) for a given prompt. Therefore, prompts that the model can answer more correctly, also show stronger within-sample correlations. This result is similar to Finding 4 which notes that correctness approximately increases with accuracy.

Self-consistency. Self-consistency (SC) aggregates multiple samples via majority voting (using uniform or probability weighting) on the extracted answers (Wang et al., 2023). Typically, uniform weighting improves upon probability-based weighting. This can be explained by Finding 5 which states that there is no particular correlation by repeatedly drawing multiple samples for a prompt. See Appendix E.2 for more details and ablations. We summarize our results in Finding 7.

Finding 6: For low-temperature next-token sampling, self-consistency with majority voting outperforms probability weighting. One reason might be that the within-sample correlation is distributed symmetrically around zero (positive and negative correlations in equal parts).

4 Discussion and Conclusion

Our results show that the relationship between sequence probability and correctness depends on the level it is measured at. Within a fixed dataset, model, and decoding setup, higher-probability prompt-answer pairs are often more likely to be correct, suggesting that sequence probability contains useful information across examples. However, this signal does not generally transfer to the method-level. When comparing responses to the same prompt, tuning hyperparameters within a method, or comparing different decoding methods, higher sequence probability does not consistently imply higher correctness.

Our analysis has practical implications for probability-based decoding, self-consistency, and probability-based verifier-free self-improvement. Specifically, sequence probabilities are a useful signal, within a given dataset and across samples for a single prompt, when the model already has sufficient task accuracy. However, it is a less useful criterion for selecting hyperparameters of methods and methods themselves. Therefore, our findings suggest that in addition to asking *whether* sequence probability correlates with correctness, it is important to ask *at which granularity* it does so.

REFERENCES

Seyedarmin Azizi, Erfan Baghaei Potraghloo, Minoo Ahmadi, Souvik Kundu, and Massoud Pedram. Power-smc: Low-latency sequence-level power sampling for training-free llm reasoning. *arXiv preprint arXiv:2602.10273*, 2026. Cited on pages 1, 4, 5, 14, 18, and 20.

Mark Briers, Arnaud Doucet, and Simon Maskell. Smoothing algorithms for state–space models. *Annals of the Institute of Statistical Mathematics*, 62:61–89, 2010. Cited on pages 14 and 18.

Haolin Chen, Yihao Feng, Zuxin Liu, Weiran Yao, Akshara Prabhakar, Shelby Heinecke, Ricky Ho, Phil Mui, Silvio Savarese, Caiming Xiong, et al. Language models are hidden reasoners: Unlocking latent reasoning capabilities via self-rewarding. *arXiv preprint arXiv:2411.04282*, 2024a. Cited on page 38.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code, 2021. *arXiv preprint arXiv:2107.03374*, 2025. Cited on page 5.

Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. Universal self-consistency for large language models. In *International Conference on Machine Learning, Workshop on In-Context Learning*, 2024b. Cited on page 14.

Nicolas Chopin. Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference. *The Annals of Statistics*, 32(6):2385 – 2411, 2004. Cited on page 18.

Nicolas Chopin, Omiros Papaspiliopoulos, et al. *An introduction to sequential Monte Carlo*, volume 4. Springer, 2020. Cited on pages 14 and 18.

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. *arXiv preprint arXiv:2110.14168*, 2021. Cited on page 14.

Eldan Cohen and Christopher Beck. Empirical analysis of beam search performance degradation in neural sequence models. In *International Conference on Machine Learning*, 2019. Cited on pages 1, 3, and 14.

Pierre Del Moral. *Feynman-Kac Formulae*, pp. 47–93. Springer New York, 2004. Cited on page 18.

Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. Sequential monte carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436, 2006. Cited on pages 14 and 18.

Arnaud Doucet, Nando De Freitas, and Neil Gordon. An introduction to sequential monte carlo methods. *Sequential Monte Carlo methods in practice*, pp. 3–14, 2001. Cited on pages 14 and 18.

Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical Neural Story Generation. In Iryna Gurevych and Yusuke Miyao (eds.), *Annual Meeting of the Association for Computational Linguistics, Long Papers*, 2018. Cited on pages 1, 3, and 14.

Yichao Fu, Xuewei Wang, Yuandong Tian, and Jiawei Zhao. Deep Think with Confidence. *arXiv preprint arXiv:2508.15260*, 2025. Cited on page 14.

J. Gai, G. Zeng, H. Zhang, and A. Raghunathan. Differential smoothing mitigates sharpening and improves LLM reasoning. *arXiv preprint arXiv:2511.19942*, 2025. Cited on page 14.

W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970. Cited on page 14.

Amy W. He, Daniel Fried, and Sean Welleck. Rewarding the unlikely: Lifting GRPO beyond distribution sharpening. In *Conference on Empirical Methods in Natural Language Processing*, 2025. Cited on page 14.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021. Cited on page 5.

John Hewitt, Christopher D. Manning, and Percy Liang. Truncation Sampling as Language Model Desmoothing. *arXiv preprint arXiv:2210.15191*, 2022. Cited on pages 3 and 14.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. Cited on pages 1 and 38.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The Curious Case of Neural Text Degeneration. *arXiv preprint arXiv:1904.09751*, 2020. Cited on pages 3 and 14.

Xiaotong Ji, Rasul Tutunov, Matthieu Zimmer, and Haitham Bou Ammar. Scalable power sampling: Unlocking efficient, training-free reasoning for llms via distribution sharpening. *arXiv preprint arXiv:2601.21590*, 2026. Cited on pages 1, 4, 5, 14, 18, and 20.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421, 2021. Cited on page 5.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022. Cited on page 14.

Zhewei Kang, Xuandong Zhao, and Dawn Song. Scalable Best-of-N Selection for Large Language Models via Self-Certainty. *arXiv preprint arXiv:2502.18581*, 2025. Cited on page 14.

Aayush Karan and Yilun Du. Reasoning with sampling: Your base model is smarter than you think. *arXiv preprint arXiv:2510.14901*, 2025. Cited on pages 1, 4, 5, and 14.

Tom Kempton and Stuart Burrell. Local normalization distortion and the thermodynamic formalism of decoding strategies for large language models. *arXiv preprint arXiv:2503.21929*, 2025. Cited on page 14.

Philipp Koehn and Rebecca Knowles. Six Challenges for Neural Machine Translation. In *Annual Meeting of the Association for Computational Linguistics, Workshop on Neural Machine Translation*, 2017. Cited on pages 1, 3, and 14.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Symposium on Operating Systems Principles*, 2023. Cited on pages 4 and 20.

Dieterich Lawson, Allan Raventós, Andrew Warrington, and Scott Linderman. Sixo: Smoothing inference with twisted objectives. In *Advances in Neural Information Processing Systems*, 2022. Cited on page 19.

Xueyan Li, Guinan Su, Mrinmaya Sachan, and Jonas Geiping. Sample smart, not hard: Correctness-first decoding for better reasoning in LLMs. In *International Conference on Learning Representations*, 2026a. Cited on page 14.

Xueyan Li, Johannes Zenn, Ekaterina Fadeeva, Guinan Su, Mrinmaya Sachan, and Jonas Geiping. ε-leaf enumeration: Non-repeating self-consistency via truncated tree search. *International Conference on Learning Representations (ICLR), Workshop on Latent & Implicit Thinking*, 2026b. Cited on page 14.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024. Cited on page 14.

Vasileios Lioutas, Jonathan Wilder Lavington, Justice Sefas, Matthew Niedoba, Yunpeng Liu, Berend Zwartsenberg, Setareh Dabiri, Frank Wood, and Adam Scibior. Critic sequential monte carlo. In *International Conference on Learning Representations*, 2022. Cited on page 19.

Bruce T Lowerre. *The harpy speech recognition system.* Carnegie Mellon University, 1976. Cited on pages 1, 3, and 14.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953. Cited on page 14.

K. Ni, Z. Tan, Z. Liu, P. Li, and T. Chen. Can GRPO help LLMs transcend their pretraining origin? *arXiv preprint arXiv:2510.15990*, 2025. Cited on page 14.

Karl Pearson. Iii. contributions to the mathematical theory of evolution. *Philosophical Transactions of the Royal Society of London. (A.)*, (185):71–110, 12 1894. Cited on page 20.

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 q&a benchmark. In *Conference on Language Modeling*, 2024. Cited on page 5.

Y. Song, J. Kempe, and R. Munos. Outcome-based exploration for LLM reasoning. *arXiv preprint arXiv:2509.06941*, 2025. Cited on page 14.

C. Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101, 1904. ISSN 00029556. URL http://www.jstor.org/stable/1412159. Cited on page 20.

Claudio Spiess, David Gros, Kunal Suresh Pai, Michael Pradel, Md Rafiqul Islam Rabin, Amin Alipour, Susmit Jha, Prem Devanbu, and Toufique Ahmed. Calibration and correctness of language models for code. In *International Conference on Software Engineering*, 2025. Cited on page 14.

Yixuan Su, Tian Lan, Yan Wang, Dani Yogatama, Lingpeng Kong, and Nigel Collier. A contrastive framework for neural text generation. In *Advances in Neural Information Processing Systems*, 2022. Cited on pages 1, 3, and 14.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 2014. Cited on pages 1, 3, and 14.

Amir Taubenfeld, Tom Sheffer, Eran Ofek, Amir Feder, Ariel Goldstein, Zorik Gekhman, and Gal Yona. Confidence improves self-consistency in LLMs. In *Findings of the Association for Computational Linguistics*, 2025. Cited on page 1.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*, 2016. Cited on pages 1, 3, and 14.

Luke Vilnis, Yury Zemlyanskiy, Patrick Murray, Alexandre Tachard Passos, and Sumit Sanghai. Arithmetic sampling: parallel diverse decoding for large language models. In *International Conference on Machine Learning*, 2023. Cited on page 14.

Han Wang, Archiki Prasad, Elias Stengel-Eskin, and Mohit Bansal. Soft Self-Consistency Improves Language Models Agents. In *Annual Meeting of the Association for Computational Linguistics, Short Papers*, 2024a. Cited on pages 1 and 14.

Xinglin Wang, Yiwei Li, Shaoxiong Feng, Peiwen Yuan, Boyuan Pan, Heda Wang, Yao Hu, and Kan Li. Integrate the essence and eliminate the dross: Fine-grained self-consistency for free-form language generation. In *Annual Meeting of the Association for Computational Linguistics, Long Papers*, 2024b. Cited on page 14.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-Consistency Improves Chain of Thought Reasoning in Language Models. *arXiv preprint arXiv:2203.11171*, 2023. Cited on pages 1, 9, 14, and 23.

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. *arXiv preprint arXiv:2406.01574*, 2024c. Cited on pages 5 and 21.

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. *arXiv preprint arXiv:2505.09388*, 2025. Cited on page 5.

Yilin Yang, Liang Huang, and Mingbo Ma. Breaking the Beam Search Curse: A Study of (Re-)Scoring Methods and Stopping Criteria for Neural Machine Translation. In *Conference on Empirical Methods in Natural Language Processing*, 2018. Cited on pages 1, 3, and 14.

Y. Yue, Z. Chen, R. Lu, A. Zhao, Z. Wang, S. Song, and G. Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025. Cited on page 14.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*, 2022. Cited on page 38.

Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *International Conference on Computer Vision*, 2019. Cited on pages 1 and 38.

Stephen Zhao, Rob Brekelmans, Alireza Makhzani, and Roger Baker Grosse. Probabilistic inference in language models via twisted sequential monte carlo. In *International Conference on Machine Learning*, 2024. Cited on pages 18 and 19.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023. Cited on page 5.

A Related Work

In this section, we give a broad overview over the related work.

Decoding methods. Search-based decoding methods methods like beam search (Lowerre, 1976; Sutskever et al., 2014) searches over the space of sequences but often suffers from repetitions (Vijayakumar et al., 2016; Cohen & Beck, 2019), and other deteriorations in quality (Koehn & Knowles, 2017; Yang et al., 2018). Its lack of diversity (Su et al., 2022) has been addressed by adding terms related to diversity (Vijayakumar et al., 2016; Vilnis et al., 2023). In Section 2, we discuss various local decoding methods: top-k sampling (Fan et al., 2018), top-p sampling (Holtzman et al., 2020), and ε-sampling (Hewitt et al., 2022). Kempton & Burrell (2025) phrase common local decoding methods in a framework similar to our variational objective. In this work, we instead consider a variational framework for global decoding methods. While most prior work investigate how decoding relates to accuracy, this work asks the more general question how sequence probability and correctness are related.

Power sampling in LLMs. Karan & Du (2025) introduce power sampling from LLMs as a verifier-free way to sharpen the model distribution over full sequences. They propose a Markov Chain Monte Carlo algorithm, i.e., a Metropolis-Hastings (Metropolis et al., 1953; Hastings, 1970) sampler operating in blocks of tokens. Ji et al. (2026) take a similar block-based framework and approximate the token-level power distribution by using online rollouts from the top-k tokens resulting in the scalable power sampling (SPS) algorithm. Azizi et al. (2026) propose power-SMC, a sequential Monte Carlo (SMC) (Doucet et al., 2001; Del Moral et al., 2006; Briers et al., 2010; Chopin et al., 2020) sampler, which moves a set of particles over T steps (where T denotes the length of the sequence), computes importance-weights, and resamples the particles. In this work, we ask the more general question on how sequence log-probability and correctness are related. We investigate sampling from the power distribution as one global method producing more probable sequences (see Remark 2.2) approximated by SPS and power-SMC.

The power distribution, sharpening, and reinforcement learning. Many works argue that reinforcement learning with verifiable rewards (RLVR) can be explained as doing distribution sharpening (He et al., 2025; Song et al., 2025; Yue et al., 2025; Gai et al., 2025; Ni et al., 2025). Based on this Karan & Du (2025) make the point that power sampling can be compared to RLVR and show comparisons between RL-finetuned models and base models. We, instead, show results for base and posttrained models and focus on the relationship between sequence probability and correctness instead of explicitly comparing models at different training stages.

Self-consistency. Self-consistency (SC) aggregates multiple samples from a LLM via majority voting on extracted answers (Wang et al., 2023). In cases where (simple) answer extraction is not possible, an LLM can be used to extract and compare answers (Wang et al., 2024a; Chen et al., 2024b; Wang et al., 2024b). In this work, we analyze the correlation between probability and correctness, also *within a sample*. We find that, often, there is no consistent correlation for a single prompt, explaining why probability-weighting in SC often underperforms.

Choosing out of a pool of N sequences. Best-of-N (Cobbe et al., 2021; Lightman et al., 2024) is a general method choosing the "best" sequence out of a set of N sequences. While sequences can be chosen by probability, various other confidence criteria can be defined (Kang et al., 2025; Fu et al., 2025). These criteria encode assumption about how correct sequences look like. In this work, we investigate the underlying question asking how log-probability and correctness are related.

Calibration and log-probability. There are various works investigating calibration in LLMs. Spiess et al. (2025) focus on the calibration of LLMs for code. They find that code models are not well calibrated in general and they show how calibration can be improved. Kadavath et al. (2022) show that LLMs are often well-calibrated on multiple-choice questions and questions with binary options. Then, they evaluate self-evaluation by asking models for answers and letting them evaluate the probability of the answer being correct. This work is related to calibration in that it similarly computes the log-probability for sequences and compares it to its accuracy (within-dataset correlation). However, we do not relate the specific sequence-probabilities to correctness and consider various other forms of correlation. Li et al. (2026b) motivate their tree-search method by showing that *coverage* i.e., the log-probability of all samples generated for a given prompt, correlates with correctness. Further, Li

et al. (2026a) find that *low probability tokens* correlate with large *epistemic uncertainty* and propose to resort to greedily sample those tokens (as opposed to sampling from the distribution).

B THEORETICAL ARGUMENTS

In this section, we provide an extended formal discussion on decoding methods highlighting various connections made in the main text.

First, we can state that beam search (without length normalization) becomes exact with large K.

Remark 2.1 (Beam search becomes exact for $K \to \infty$)**.** *If K is chosen such that no continuation is ever pruned before a fixed T then beam search returns an element of* $\arg\max_{\mathbf{s}} p(\mathbf{s} \mid \bar{\mathbf{s}})$.

Proof. For a fixed horizon T, beam search keeps the K highest-scoring partial continuations at each depth. If K is large enough that such that no continuation of length at most T is pruned, then beam search explores the entire tree up to depth T. Hence, at depth T, it compares all complete continuations $\mathbf{s}$ and returns one with maximal probability under $p(\mathbf{s} \mid \bar{\mathbf{s}})$. □

Power sampling, another global sampling method, increases the *expected* sequence log-probability.

Remark 2.2 (Power sampling increases expected sequence log-probability)**.** *For $\alpha > 1$, let $\mathbf{s} \sim p_\alpha(\cdot \mid \bar{\mathbf{s}})$. Then, increasing α shifts mass toward sequences with larger probability under p as* $\mathbb{E}_{p_\alpha}[\log p(\mathbf{s} \mid \bar{\mathbf{s}})] \geq \mathbb{E}_p[\log p(\mathbf{s} \mid \bar{\mathbf{s}})]$.

Proof. Let

$$Z_\alpha(\bar{\mathbf{s}}) = \sum_{\mathbf{s}} p(\mathbf{s} \mid \bar{\mathbf{s}})^\alpha.$$

Then, under reasonable assumptions,

$$\frac{d}{d\alpha} \log Z_\alpha(\bar{\mathbf{s}}) = \frac{\sum_{\mathbf{s}} p(\mathbf{s} \mid \bar{\mathbf{s}})^\alpha \log p(\mathbf{s} \mid \bar{\mathbf{s}})}{Z_\alpha(\bar{\mathbf{s}})} = \mathbb{E}_{p_\alpha(\mathbf{s}|\bar{\mathbf{s}})}[\log p(\mathbf{s} \mid \bar{\mathbf{s}})].$$

Differentiating once more gives

$$\frac{d}{d\alpha} \mathbb{E}_{p_\alpha(\mathbf{s}|\bar{\mathbf{s}})}[\log p(\mathbf{s} \mid \bar{\mathbf{s}})] = \mathrm{Var}_{p_\alpha(\mathbf{s}|\bar{\mathbf{s}})}(\log p(\mathbf{s} \mid \bar{\mathbf{s}})) \geq 0.$$

Hence $\mathbb{E}_{p_\alpha(\mathbf{s}|\bar{\mathbf{s}})}[\log p(\mathbf{s} \mid \bar{\mathbf{s}})]$ is nondecreasing in α. □

In contrast, for *local* sampling methods, we cannot guarantee any *global* properties.

Remark B.1 (Local sampling methods do not target most probable global sequences.)**.** *Methods with local normalization constants modify $p(\cdot \mid \bar{\mathbf{s}}, \mathbf{s}_{<t})$ separately at each prefix. Hence, they favor continuations whose* next token *is locally likely. This is different from maximizing the* sequence-level *probability $p(\mathbf{s} \mid \bar{\mathbf{s}})$.*

Proof. For LTS,

$$\tilde{p}_\alpha(\mathbf{s} \mid \bar{\mathbf{s}}) = \prod_{t=1}^{T} \frac{p(s_t \mid \bar{\mathbf{s}}, \mathbf{s}_{<t})^\alpha}{\tilde{Z}_\alpha(\bar{\mathbf{s}}, \mathbf{s}_{<t})} = p(\mathbf{s} \mid \bar{\mathbf{s}})^\alpha \prod_{t=1}^{T} \tilde{Z}_\alpha(\bar{\mathbf{s}}, \mathbf{s}_{<t})^{-1}. \tag{B1}$$

For TS,

$$p_A(\mathbf{s} \mid \bar{\mathbf{s}}) = \prod_{t=1}^{T} \frac{p(s_t \mid \bar{\mathbf{s}}, \mathbf{s}_{<t})}{\tilde{Z}_A(\bar{\mathbf{s}}, \mathbf{s}_{<t})} \mathbb{1}[s_t \in A(\bar{\mathbf{s}}, \mathbf{s}_{<t})] = p(\mathbf{s} \mid \bar{\mathbf{s}}) \prod_{t=1}^{T} \frac{\mathbb{1}[s_t \in A(\bar{\mathbf{s}}, \mathbf{s}_{<t})]}{\tilde{Z}_A(\bar{\mathbf{s}}, \mathbf{s}_{<t})}. \tag{B2}$$

In both, Equation (B1) and Equation (B2), the sequence weight contains factors beyond $p(\mathbf{s} \mid \bar{\mathbf{s}})$ itself. This is different from the normalization constant $Z_\alpha(\bar{\mathbf{s}}) = \sum_{\mathbf{s}} p(\mathbf{s} \mid \bar{\mathbf{s}})^\alpha$ (Equation (4) in Section 2). □

LTS defines a temperature α similar to power-sampling. However, there is, in general, no temperature for LTS that recovers samples from the power distribution.

Remark 2.3 (LTS is generally not equal to power sampling.)**.** *Unless the local normalization constants of LTS are identical across all reachable prefixes, no single token-level temperature reproduces the sequence-level power distribution.*

Proof. The LTS-induced sequence distribution is

$$\tilde{p}_\alpha(\mathbf{s} \mid \bar{\mathbf{s}}) = \prod_{t=1}^{T} \frac{p(s_t \mid \bar{\mathbf{s}}, \mathbf{s}_{<t})^\alpha}{\tilde{Z}_\alpha(\bar{\mathbf{s}}, \mathbf{s}_{<t})} = \frac{p(\mathbf{s} \mid \bar{\mathbf{s}})^\alpha}{\prod_{t=1}^{T} \tilde{Z}_\alpha(\bar{\mathbf{s}}, \mathbf{s}_{<t})}. \tag{B3}$$

By contrast, the sequence-level power distribution is

$$p_\alpha(\mathbf{s} \mid \bar{\mathbf{s}}) = \frac{p(\mathbf{s} \mid \bar{\mathbf{s}})^\alpha}{Z_\alpha(\bar{\mathbf{s}})}.$$

Thus LTS coincides with sequence-level power sampling only if

$$\prod_{t=1}^{T} \tilde{Z}_\alpha(\bar{\mathbf{s}}, \mathbf{s}_{<t}) \overset{!}{=} Z_\alpha(\bar{\mathbf{s}}) \;\forall\; (\bar{\mathbf{s}}, \mathbf{s}_{<t}).$$

However, the local normalization constants depend on the prefix, so the product is path-dependent and LTS is not equal to sequence-level power sampling. □

As discussed in the main test, we find the power distribution to be the maximizer of $\mathcal{J}_\alpha(q)$.

Remark 2.4 (The power distribution is a maximizer of the variational objective.)**.** *The maximizer of $\mathcal{J}_\alpha(q)$ is the power distribution, $q^\star_\alpha \in \arg\max_q \mathcal{J}_\alpha(q) \iff q^\star_\alpha(\mathbf{s} \mid \bar{\mathbf{s}}) = p_\alpha(\mathbf{s} \mid \bar{\mathbf{s}})$.*

Proof. We have $\mathcal{J}_\alpha(q) = \sum_{\mathbf{s}} q(\mathbf{s} \mid \bar{\mathbf{s}}) \log p(\mathbf{s} \mid \bar{\mathbf{s}}) - \frac{1}{\alpha} \sum_{\mathbf{s}} q(\mathbf{s} \mid \bar{\mathbf{s}}) \log q(\mathbf{s} \mid \bar{\mathbf{s}})$, and,

$$\mathcal{J}_\alpha(q) = -\frac{1}{\alpha} \sum_{\mathbf{s}} q(\mathbf{s} \mid \bar{\mathbf{s}}) \log \frac{q(\mathbf{s} \mid \bar{\mathbf{s}})}{p(\mathbf{s} \mid \bar{\mathbf{s}})^\alpha} \tag{B4}$$

$$= -\frac{1}{\alpha} \sum_{\mathbf{s}} q(\mathbf{s} \mid \bar{\mathbf{s}}) \log \frac{q(\mathbf{s} \mid \bar{\mathbf{s}})}{p_\alpha(\mathbf{s} \mid \bar{\mathbf{s}}) Z_\alpha(\bar{\mathbf{s}})} \tag{B5}$$

$$= -\frac{1}{\alpha} D_{\mathrm{KL}}\big(q(\cdot \mid \bar{\mathbf{s}}) \,\|\, p_\alpha(\cdot \mid \bar{\mathbf{s}})\big) + \frac{1}{\alpha} \log Z_\alpha(\bar{\mathbf{s}}), \tag{B6}$$

where

$$Z_\alpha(\bar{\mathbf{s}}) = \sum_{\mathbf{s}} p(\mathbf{s} \mid \bar{\mathbf{s}})^\alpha.$$

Since $\log Z_\alpha(\bar{\mathbf{s}})$ does not depend on q, maximizing $\mathcal{J}_\alpha(q)$ is equivalent to minimizing

$$D_{\mathrm{KL}}\big(q(\cdot \mid \bar{\mathbf{s}}) \,\|\, p_\alpha(\cdot \mid \bar{\mathbf{s}})\big),$$

whose unique minimizer is $q(\cdot \mid \bar{\mathbf{s}}) = p_\alpha(\cdot \mid \bar{\mathbf{s}})$. □

Investigating the hyperparameter α, we find a tradeoff between expected log-probability and entropy.

Remark B.2 (Expected log-probability increases and entropy decreases with α)**.** *For $\alpha > 1$, increasing α shifts mass toward higher-probability continuations under p. Therefore, $\mathbb{E}_{p_\alpha}[\log p(\mathbf{s} \mid \bar{\mathbf{s}})]$ is monotonically increasing in α, while $\mathcal{H}(p_\alpha)$ is monotonically decreasing in α.*

Proof. We have seen the first part already in Remark 2.2. For the entropy, we obtain

$$\mathcal{H}(p_\alpha) = -\sum_{\mathbf{s}} p_\alpha(\mathbf{s} \mid \bar{\mathbf{s}}) \log p_\alpha(\mathbf{s} \mid \bar{\mathbf{s}}) \tag{B7}$$

$$= -\sum_{\mathbf{s}} p_\alpha(\mathbf{s} \mid \bar{\mathbf{s}}) \Big(\alpha \log p(\mathbf{s} \mid \bar{\mathbf{s}}) - \log Z_\alpha(\bar{\mathbf{s}})\Big) \tag{B8}$$

$$= -\alpha\, \mathbb{E}_{p_\alpha}[\log p(\mathbf{s} \mid \bar{\mathbf{s}})] + \log Z_\alpha(\bar{\mathbf{s}}). \tag{B9}$$

Differentiating gives

$$\frac{\partial}{\partial \alpha} \mathcal{H}(p_\alpha) = -\alpha \frac{\partial}{\partial \alpha} \mathbb{E}_{p_\alpha}[\log p(\mathbf{s} \mid \bar{\mathbf{s}})] \tag{B10}$$

$$= -\alpha \operatorname{Var}_{p_\alpha}(\log p(\mathbf{s} \mid \bar{\mathbf{s}})) \le 0. \tag{B11}$$

Therefore, $\mathcal{H}(p_\alpha)$ is monotonically decreasing in α. □

C A TWISTED SEQUENTIAL MONTE CARLO FRAMEWORK FOR POWER SAMPLING

Sequential Monte Carlo (SMC) algorithms (Doucet et al., 2001; Del Moral et al., 2006; Briers et al., 2010; Chopin et al., 2020) are one of the main algorithms for estimating (global) normalization constants. In this section, we show how recent work (Ji et al., 2026; Azizi et al., 2026) fits into the *twisted SMC framework* of Zhao et al. (2024).

Let $\gamma(\mathbf{s} \mid \bar{\mathbf{s}}) \coloneqq p(\mathbf{s} \mid \bar{\mathbf{s}})^\alpha$ denote the target distribution. Since exact sampling from the normalized distribution induced by γ is intractable, a natural approach is (twisted) SMC which constructs a sequence intermediate target distributions and propagates a population of particles. Twisted SMC (Zhao et al., 2024) introduces positive twist functions $\psi_t(\bar{\mathbf{s}}, \mathbf{s}_{\leq t})$ and defines $\gamma_t(\mathbf{s}_{\leq t} \mid \bar{\mathbf{s}}) \coloneqq p(\mathbf{s}_{\leq t} \mid \bar{\mathbf{s}})^\alpha \, \psi_t(\bar{\mathbf{s}}, \mathbf{s}_{\leq t})$ for $t < T$, with $\gamma_T = \gamma$. For a proposal $q_t(s_t \mid \bar{\mathbf{s}}, \mathbf{s}_{<t})$, the incremental weight is $w_t^i \propto w_{t-1}^i p(s_t^i \mid \bar{\mathbf{s}}, \mathbf{s}_{<t}^i)^\alpha / q_t(s_t^i \mid \bar{\mathbf{s}}, \mathbf{s}_{<t}^i) \psi_t(\bar{\mathbf{s}}, \mathbf{s}_{\leq t}^i) / \psi_{t-1}(\bar{\mathbf{s}}, \mathbf{s}_{<t}^i)$.

Remark C.1 (Optimal twist following Zhao et al. (2024, Proposition 3.2))**.** *The optimal twist functions targeting the power distribution are given as*

$$\psi_t^\star(\bar{\mathbf{s}}, \mathbf{s}_{\leq t}) \propto \sum_{\mathbf{s}_{>t}} p(\mathbf{s}_{>t} \mid \bar{\mathbf{s}}, \mathbf{s}_{\leq t})^\alpha. \tag{C1}$$

Proof. The terminal unnormalized target is

$$\gamma_T(\mathbf{s} \mid \bar{\mathbf{s}}) = p(\mathbf{s} \mid \bar{\mathbf{s}})^\alpha.$$

For a fixed prefix $\mathbf{s}_{\leq t}$, summing out the future suffix gives

$$\sum_{\mathbf{s}_{>t}} \gamma_T(\mathbf{s} \mid \bar{\mathbf{s}}) = \sum_{\mathbf{s}_{>t}} \Big(p(\mathbf{s}_{\leq t} \mid \bar{\mathbf{s}})^\alpha p(\mathbf{s}_{>t} \mid \bar{\mathbf{s}}, \mathbf{s}_{\leq t})^\alpha \Big) \tag{C2}$$

$$= p(\mathbf{s}_{\leq t} \mid \bar{\mathbf{s}})^\alpha \sum_{\mathbf{s}_{>t}} p(\mathbf{s}_{>t} \mid \bar{\mathbf{s}}, \mathbf{s}_{\leq t})^\alpha. \tag{C3}$$

Therefore, choosing

$$\psi_t^\star(\bar{\mathbf{s}}, \mathbf{s}_{\leq t}) \propto \sum_{\mathbf{s}_{>t}} p(\mathbf{s}_{>t} \mid \bar{\mathbf{s}}, \mathbf{s}_{\leq t})^\alpha$$

makes

$$\gamma_t(\mathbf{s}_{\leq t} \mid \bar{\mathbf{s}}) = p(\mathbf{s}_{\leq t} \mid \bar{\mathbf{s}})^\alpha \psi_t^\star(\bar{\mathbf{s}}, \mathbf{s}_{\leq t})$$

proportional to the exact marginal of γ_T at time t. □

The two recently proposed methods fall into this framework as we explain next.

Power-SMC. Power-SMC (Azizi et al., 2026) corresponds to the *untwisted* choice $\psi_t \equiv 1$ using the LTS proposal distribution $q_t(s_t \mid \bar{\mathbf{s}}, \mathbf{s}_{<t}) = \tilde{p}_\alpha(s_t \mid \bar{\mathbf{s}}, \mathbf{s}_{<t})$. While this minimizes the variance among prefix-only proposals (Azizi et al., 2026), it does not account for future continuation mass.

Remark C.2 (Asymptotic exactness of Power-SMC following Chopin (2004) and Del Moral (2004).)**.** *For any fixed horizon, consistency of SMC implies that, under the usual regularity assumptions, the empirical particle approximation converges to the target distribution as $N \to \infty$.*

Proof. Power-SMC is a standard SMC algorithm for the Feynman-Kac model with unnormalized targets

$$\gamma_t(\mathbf{s}_{\leq t} \mid \bar{\mathbf{s}}) = p(\mathbf{s}_{\leq t} \mid \bar{\mathbf{s}})^\alpha$$

and proposal

$$q_t(s_t \mid \bar{\mathbf{s}}, \mathbf{s}_{<t}) = \tilde{p}_\alpha(s_t \mid \bar{\mathbf{s}}, \mathbf{s}_{<t}).$$

Therefore the usual SMC law of large numbers applies. We refer to Chopin (2004) and Del Moral (2004) for the corresponding theoretical arguments. □

Scalable power sampling (SPS). Similarly, SPS (Ji et al., 2026) corresponds to a *twisted* SMC method that approximates the optimal term $\psi_t^\star(\bar{\mathbf{s}}, \mathbf{s}_{<t}, s_t)$ in Equation (C1) via Monte Carlo sampling restricted to a top-k candidate set. SPS utilizes a blockwise approximation for a block $b_t = \mathbf{s}_{(t-1)B+1:tB}$ of B tokens, $p_\alpha^{\psi_\star}(b_t \mid \bar{\mathbf{s}}, \mathbf{s}_{1:(t-1)B}) \propto p(b_t \mid \bar{\mathbf{s}}, \mathbf{s}_{1:(t-1)B})^\alpha \, \psi_{tB}^\star(\bar{\mathbf{s}}, \mathbf{s}_{1:(t-1)B} \oplus b_t)$,.

Remark C.3 (Approximation gaps for SPS)**.** *SPS is exact if the look-ahead term $\psi^\star$ is evaluated exactly at every position. Alternatively, it is exact when the candidate set is not truncated and the next block is sampled exactly from the corresponding blockwise fully-adapted conditional. In practice, the approximation comes from (i) Monte Carlo estimation of $\psi^\star$, (ii) top-k truncation, and (iii) the blockwise restriction.*

Proof. For a block b_t, the exact fully adapted block proposal is proportional to

$$p(b_t \mid \bar{\mathbf{s}}, \mathbf{s}_{1:(t-1)B})^\alpha \, \psi^\star_{tB}(\bar{\mathbf{s}}, \mathbf{s}_{1:(t-1)B} \oplus b_t).$$

If $\psi^\star$ is evaluated exactly and sampling is performed exactly from this conditional over the full candidate set, then the corresponding twisted SMC update is exact. Likewise, if no top-k truncation is applied and the next block is sampled exactly from the blockwise fully adapted conditional, then no approximation is introduced at that block update.

SPS departs from this exact construction in three places: (i) it replaces $\psi^\star$ by a Monte Carlo estimate, (ii) it truncates the candidate set to a top-k subset, and (iii) it uses a blockwise approximation rather than the exact full-sequence update. □

The twisted SMC view of Zhao et al. (2024) shows interesting directions for future work. Improvements can be mostly twofold: Firstly, one can *learn the twist functions* during inference via various methods proposed in the SMC literature (Zhao et al., 2024; Lawson et al., 2022; Lioutas et al., 2022). Orthogonally, one can extend the framework with verifiers or reward models.

D Details on Experimental Setup

In this section we discuss our implementation of sampling methods in Appendix D.1. Then we discuss the correlation coefficients in Appendix D.2, canonical hyperparameters in Appendix D.3, and our computational requirements and corresponding datasets in Appendix D.4. We provide prompt templates in Appendix D.5.

D.1 Implementation

We use most of the local sampling methods as implemented in the vLLM (Kwon et al., 2023) framework. For power-SMC we use the code as provided by the authors (Azizi et al., 2026). We implement SPS following the algorithm provided in the paper (Ji et al., 2026). We do rollouts with LTS and $\alpha = 4.0$ and correct this bias by using importance weighting to estimate the ζ functions. Further, we set $K = M = 8$ and $\tau_c = \tau_r = 0.25$ as per the original work (Ji et al., 2026). Algorithm 1 makes the exact algorithm concrete.

Algorithm 1 Importance-weighted SPS

Require: prompt $\bar{\mathbf{s}}$; base model p; candidate temperature τ_c; rollout temperature τ_r; block size B; top K; candidates per block M; power exponent α; rollouts R; rollout horizon H; max tokens T

$\mathbf{s} \leftarrow \emptyset$; $t \leftarrow 0$
while $t < T$ **and** not finished **do**
  $B_t \leftarrow \min(B,\ T - t)$
  // 1. Sample with p and temperature τ_c, score under p
  Draw $\{\tilde{\mathbf{s}}_i\}_{i=1}^{M} \sim q(\cdot \mid \bar{\mathbf{s}}, \mathbf{s})$ of $\leq B_t$ tokens where q uses p to sample with temp. τ_c
  Sort $\{\tilde{\mathbf{s}}_i\}$ by probability under p and keep top K
  // 2. Estimate ζ
  for $i = 1, \ldots, K$ **do**
    Draw $\{\mathbf{r}_{i,j}\}_{j=1}^{R} \sim q(\cdot \mid \bar{\mathbf{s}}, \mathbf{s}, \tilde{\mathbf{s}}_i)$ of $\leq H$ tokens where q uses p to sample with temp. τ_r
    $\log u_{i,j} \leftarrow \alpha \log p(\mathbf{r}_{i,j} \mid \bar{\mathbf{s}}, \mathbf{s}, \tilde{\mathbf{s}}_i) - \log q(\mathbf{r}_{i,j} \mid \bar{\mathbf{s}}, \mathbf{s}, \tilde{\mathbf{s}}_i)$
    $\log \hat{\zeta}_i \leftarrow \log\left(\frac{1}{R}\sum_j \exp(\log u_{i,j})\right)$
    for $s = 1, \ldots, R$ **do**
      $\log \hat{\zeta}_i^{(-s)} \leftarrow \log\left(\frac{1}{R-1}\sum_{j \neq s} \exp(\log u_{i,j})\right)$
    end for
  end for
  Compute the Jackknife bias correction using $\log \hat{\zeta}_i$ and $\log \hat{\zeta}_i^{(-s)}$
  // 3. Select block
  $\log b_i \leftarrow \alpha \log p(\tilde{\mathbf{s}}_i \mid \bar{\mathbf{s}}, \mathbf{s})$
  $\pi_i \propto \exp(\log b_i + \log \hat{\zeta}_i)$
  Sample $i^\star \sim \pi$; $\mathbf{s} \leftarrow \mathbf{s}|\tilde{\mathbf{s}}_i$; $t \leftarrow t + |\tilde{\mathbf{s}}_{i^\star}|$
  Mark finished if an end-of-sequence token appears in stop token was emitted in $\tilde{\mathbf{s}}_{i^\star}$
end while
return $\mathbf{s}$

D.2 Correlation Coefficients

We compute correlations between the continuous log-probability and a binary correctness label $y \in \{0, 1\}$. The per-sample Spearman rank correlation ρ (Spearman, 1904) measures monotone agreement between the two quantities. Since y is binary, the ρ is upper bound by $\sqrt{0.5}$. The binned Pearson correlation coefficient (Pearson, 1894) r partitions log-probability into $K = 10$ equal-count bins, computes the bin-mean (for log-probability and correctness) and computes the ordinary Pearson coefficient r.

D.3 Canonical Hyperparameters

Table 1 lists the canonical hyperparameters that we use in the main text.

Decoding method	Hyperparameter
LTS	$\alpha = 4$
BoN	$N = 32$
SPS	$B = 192$
Power-SMC	$N = 32$
Top-k	$k = 8$
Top-p	$p = 0.9$
Beam search	$N = 2$
ε-sampling	$\varepsilon = 0.0009$

Table 1: Canonical hyperparameters for each method.

D.4 Computational Requirements and Datasets

We set our computational budget to ≤ 1 days per method and dataset using an NVIDIA H100 graphics card with 80G of memory. To fulfill this budget, we use ≤ 500 samples of each dataset. For MMLU (Wang et al., 2024c), this corresponds to the splits "abstract algebra", "anatomy", and "astronomy". Further, we evaluate all datasets and models on a *shared set of prompts* that they generate a non-empty response for. All of these restrictions result in the numbers reported in the numbers of samples per dataset as reported in Table 2. Empty cells in our summary plots are runs that did not finish within the compute time limits.

Dataset	Number of prompts
GPQA	445
HumanEval	500
IFEval	500
MATH500	500
MedQA	500
MMLU	500

Table 2: Number of prompts per dataset for all plots.

D.5 PROMPT TEMPLATES

We use the following prompt templates for each benchmark. We prompt base models directly with the prompt and use the chat template for posttrained models.

Math500.

```
Can you solve the following math problem? {problem}
Please reason step by step, and put your final answer
within \boxed{}.
```

Humaneval.

```
Write a Python function to solve the following
problem:
{prompt}
```

GPQA.

```
Answer the following multiple choice question. The
last line of your response should be of the following
format: '\boxed{LETTER}' (without quotes) where
LETTER is one of ABCD (e.g. '\boxed{A}'). Think step
by step before answering.
{question}
A) {choice_a}
B) {choice_b}
C) {choice_c}
D) {choice_d}
```

MMLU.

```
Answer the following multiple-choice question. The
last line of your response should be of the following
format: '\boxed{LETTER}' (without quotes) where
LETTER is one of ABCD (e.g. '\boxed{A}'). Think step
by step before answering.
{question}
A) {choice_a}
B) {choice_b}
C) {choice_c}
D) {choice_d}
```

MedQA.

```
Answer the following medical multiple-choice question.
The last line of your response should be of the
following format: '\boxed{LETTER}' (without quotes)
where LETTER is one of ABCD (e.g. '\boxed{A}'). Think
step by step before answering.
{question}
A) {choice_a}
B) {choice_b}
C) {choice_c}
D) {choice_d}
```

IFEval.

```
{prompt}
```

E Additional Experiments and Analysis

In this section we provide additional experiments and analysis. Appendix E.1 shows correlation coefficients for within-dataset correlation using Spearman ρ instead of Pearson r. In , we show results on self-consistency. Appendix E.3 shows results for the model families (`Qwen2.5`, `Olmo3`) not listed in the main text. Appendix E.4 shows further and more granular results for within-sample correlation and discusses within-sample correlation results across methods. Appendix E.5 discusses log-probabilities globally, on a sequence-level, and compares log-probability to sequence-normalized log-probability. Appendix E.6 discusses sequence lengths across methods. Appendix E.7 discusses the experimental results between mode and power distribution. Appendix E.8 shows results for self-distillation. Appendix E.9 shows results for the `Qwen3` thinking models.

E.1 Computing Correlation

In this section, we compare the correlation coefficients that we use in the main text for measuring within-dataset correlation (see Section 3.4).

When comparing the per-sample Spearman correlation coefficient ρ in Figure 10 to Figure 3, we find that both plots show the same structure but Pearson correlation coefficients r seem to be more extreme.

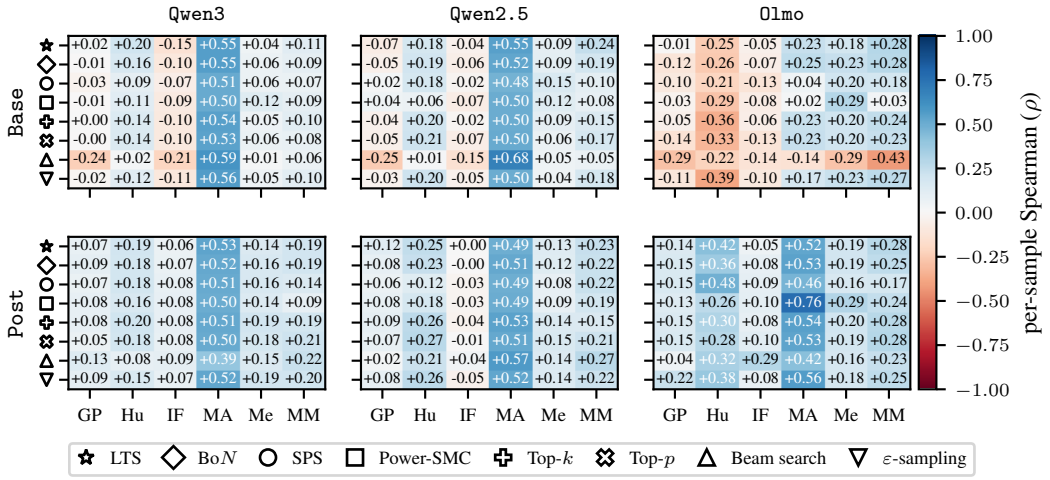


Figure 10: **We observe consistent within-dataset correlation across model families and datasets largely independent of methods.** `Base` models show more negative and diverse correlation and are consistently negative for IFEval. `Posttrained` models show consistently positive correlation. Correlation coefficients ρ averaged over model sizes at representative hyperparameter for each method. Methods are in the legend. Datasets are plotted horizontally on the bottom of each panel: GPQA, Humaneval, IFEval, MATH500, MedQA, MMLU.

E.2 Self-Consistency

Self-consistency (SC) aggregates multiple samples via majority voting (using uniform or probability weighting) on the extracted answers (Wang et al., 2023). We find that sometimes samples from the power-distribution outperform other local baselines which motivates *power* self-consistency which aggregates samples from the power-distribution. In Figure 11, we evaluate the `Qwen3` model series and we find that power self-consistency (PSC) mostly improves on MATH500 over plain self-consistency. We attribute this to the fact that power-SMC outperforms LTS most significantly on MATH500 and less so on other datasets (see Figure 5). For the other datasets, we find more diverse results, mostly degrading performance. Evaluating uniform weighting against probability weighting in Figure 12 by plotting the difference between both, we find the uniform weighting mostly degrades performance. This can be explained by Finding 5 which states that there is no particular correlation by repeatedly drawing multiple samples for a prompt. Power self-consistency in Figure 12

(right) seems to suffer less from this fact, indicating that there might be a different correlation for power-SMC. Specifically, Finding 7 makes this concrete.

Finding 7: For low-temperature next-token sampling, self-consistency with majority voting outperforms probability weighting. One reason might be that the within-sample correlation is distributed symmetrically around zero (positive and negative correlations in equal parts).

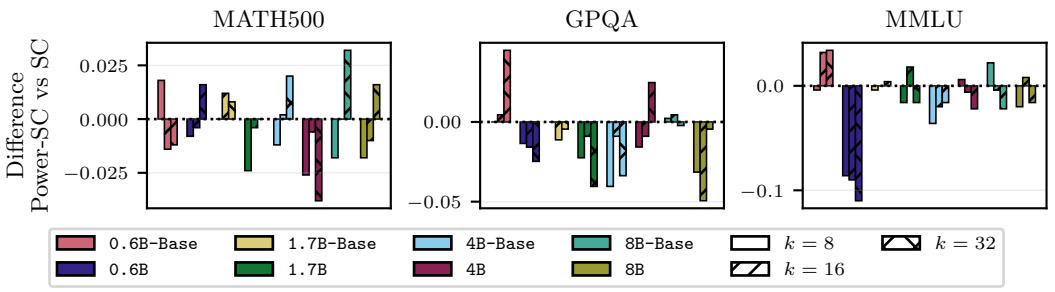


Figure 11: **Power self-consistency improves over self-consistency on MATH500.** On GPQA and MMLU power self-consistency degrades performance. Plot shows data for the `Qwen3` model series across various number of samples k. See Section 3.4 for a discussion.

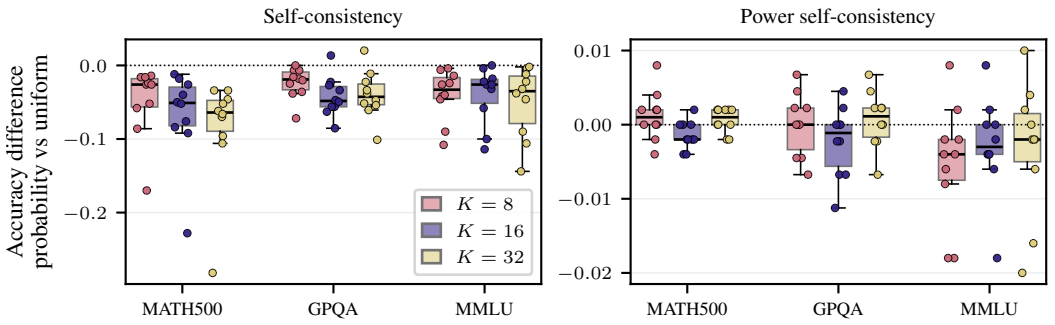


Figure 12: **Self-consistency with probability-weighted voting often underperforms majority voting. Power self-consistency does not shows consistent differences.** Accuracy difference between (power) self-consistency using uniform or probability-weighted majority voting. Results on `Qwen3` model series across three datasets. See Section 3.4 for details.

E.3 Other Models

We provide the missing models for within-method correlation (Section E.3.1) and across method correlation (Section E.3.2).

E.3.1 Within-Method Correlation

Figure 13 and Figure 14 show the within-method correlation results for `Qwen2.5` and `Olmo3`.

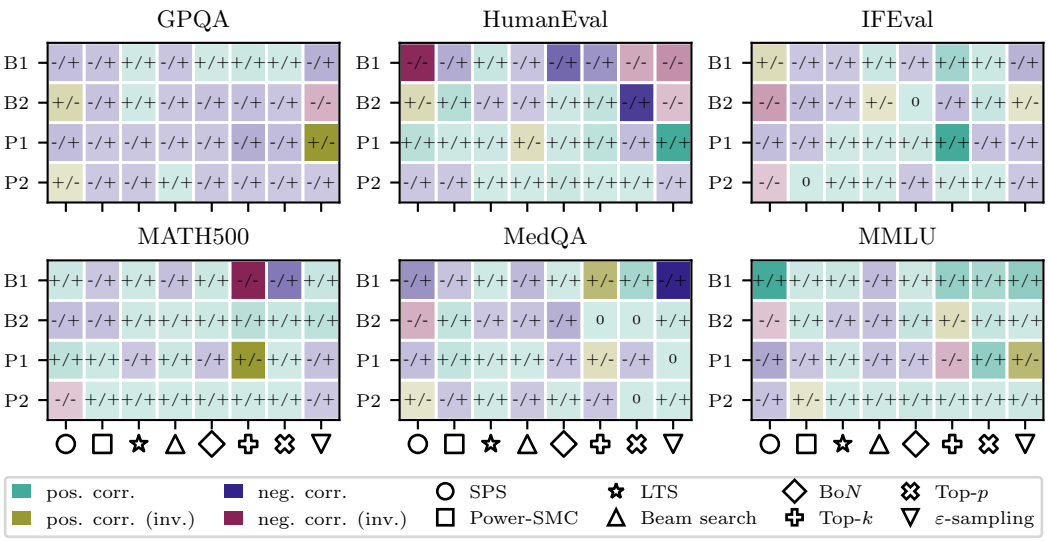


Figure 13: **Methods show either positive or negative correlation within their hyperparameter.** Correlations within methods for local and global decoding methods, models (`Qwen2.5` series 8B as ∗1, `Math` 8B as ∗2 with base as B∗ and posttrained model as P∗), and benchmark datasets. Correlations are not consistent across datasets and models. Detailed discussion in Section 3.2.

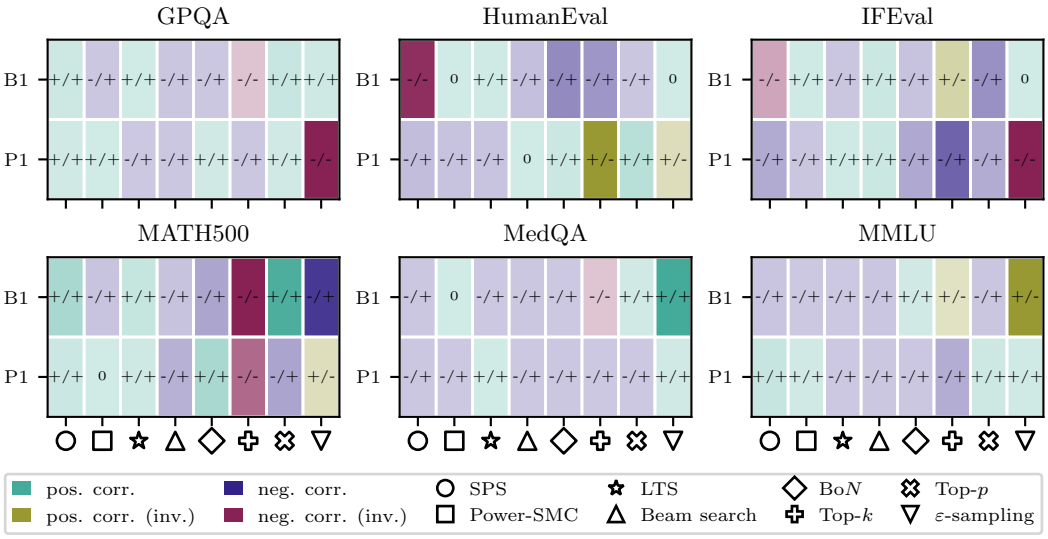


Figure 14: **Methods show either positive or negative correlation within their hyperparameter.** Correlations within methods for local and global decoding methods, models (`Olmo3` series 7B as ∗1 with base as B∗ and posttrained model as P∗), and benchmark datasets. Correlations are not consistent across datasets and models. Detailed discussion in Section 3.2.

E.3.2 Across-Method Correlation

Figure 15 and Figure 16 show the across-method correlation results for `Qwen2.5` and `Olmo3`.

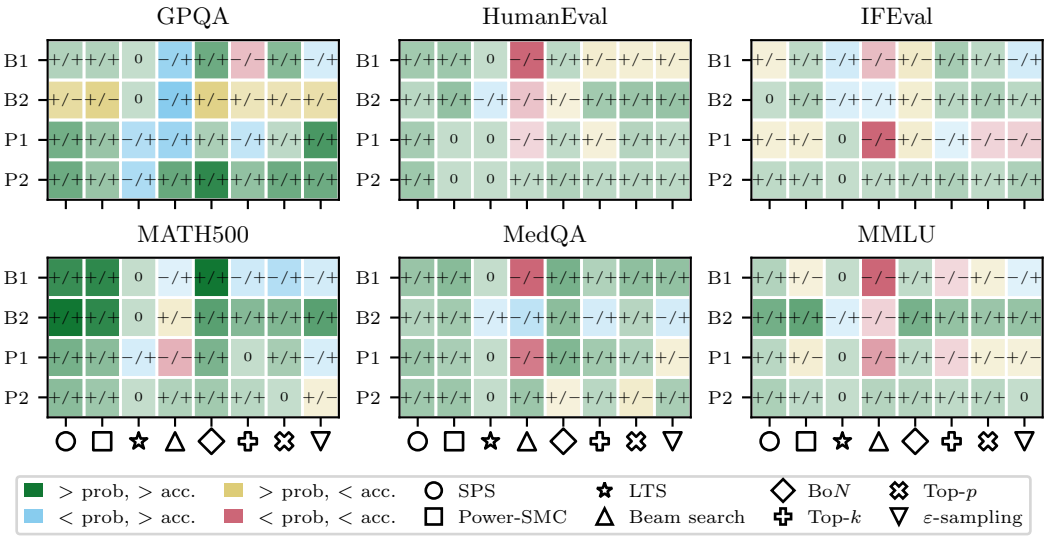


Figure 15: **Various datasets and methods show positive correlations between sequence probability and correctness.** Correlations across methods for local and global decoding methods, models (`Qwen2.5` series 8B as ∗1, `Math` 8B as ∗2 with base as B∗ and posttrained model as P∗), and benchmark datasets. Correlations are not consistent across datasets and models. See Section 3.3.

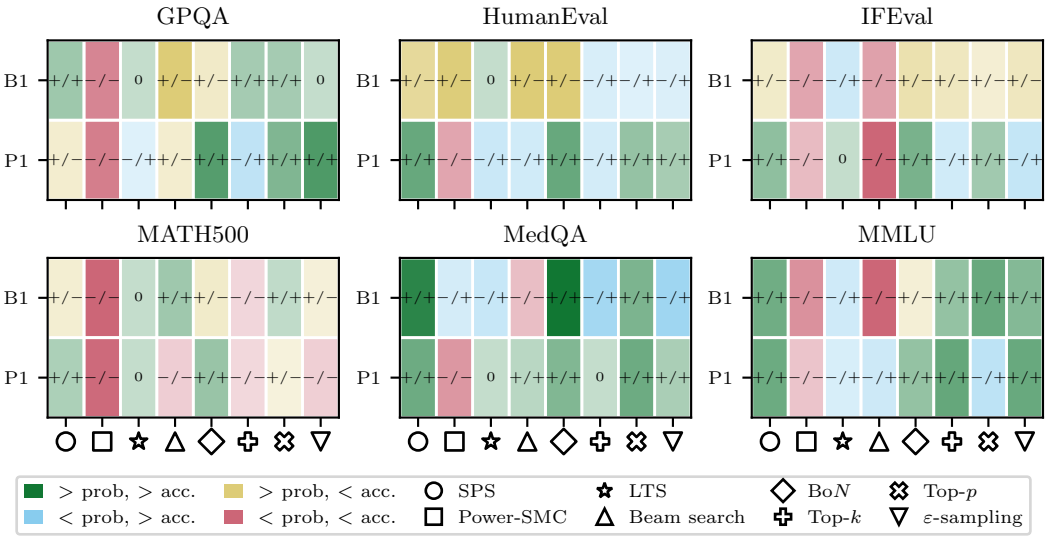


Figure 16: **Various datasets and methods show positive correlations between sequence probability and correctness.** Correlations across methods for local and global decoding methods, models(`Olmo3` series 7B as ∗1 with base as B∗ and posttrained model as P∗), and benchmark datasets. Correlations are not consistent across datasets and models. See Section 3.3.

E.4 Within-Sample Correlation

In this section we provide additional results for the within-sample correlation investigated in Section 3.4. Specifically, we investigate the results from Section 3.4 more deeply in Section E.4.1 and, further, discuss within-sample correlation across-methods in Section E.4.2.

E.4.1 Within-Sample Correlation Within Method

In this section, we extend our analysis of Section 3.4. Figure 17 shows mean correlation coefficients of Figure 9 (left, middle). We are able to consistently verify our results from the corresponding section. Specifically, we find the boxplots to summarize the granular results in Figure 17 well: While most of the correlation coefficients are of small magnitude, only for MATH500 we consistently observe positive correlation.

Moving to Figure 9 (right), we provide in Figure 18 a finer-grained analysis further splitting the data according to model family (columns) and model variants (rows). We consistently observe a similar pattern showing that for more more correct samples, also the correlation coefficient within the samples increases on average. Standard deviations are large and overlapping. The order of the means over methods changes between model families. MATH500 always seems to have highest mean.

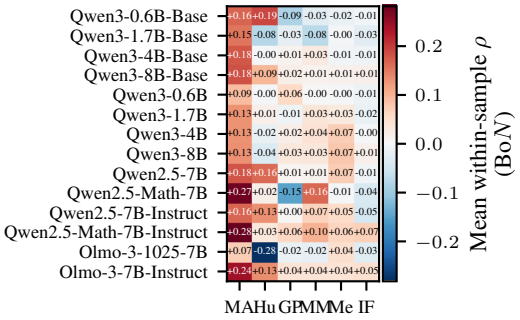


Figure 17: **Within-sample correlation coefficients are generally small with MATH500 consistently showing much larger values.** Datasets are plotted horizontally on the bottom: GPQA, Humaneval, IFEval, MATH500, MedQA, MMLU.

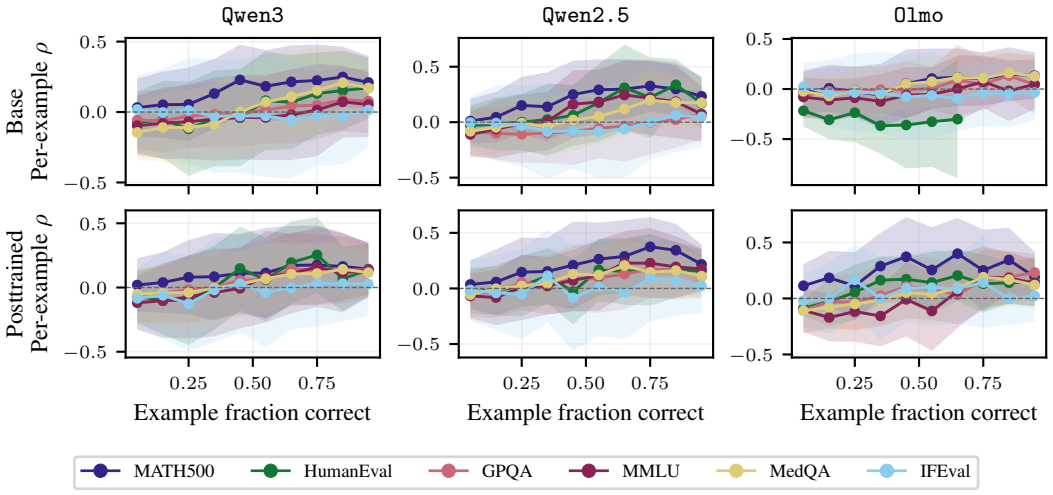


Figure 18: **The more correct a sample is under repeated sampling, the stronger its correlation coefficient.** We observe similar results across all model families (columns) and variants (rows).

E.4.2 Within-Sample Correlation Across Methods

Next, we investigate within-sample correlation across methods. To do so, we compute correlation coefficients across methods, for each sample of the dataset separately. For each method we choose a canonical hyperparameter (Table 1).

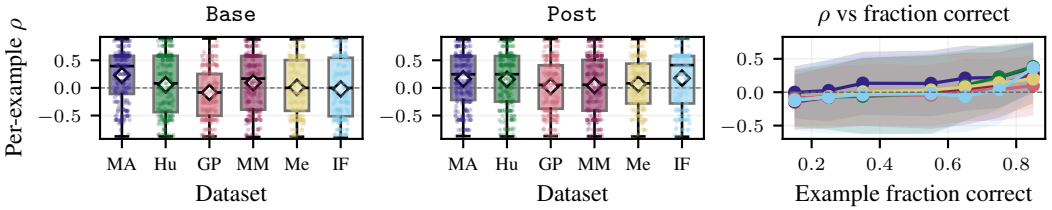


Figure 19: **Correlation coefficients are mostly symmetrically distributed around zero, the more correct a sample is, the more positive its correlation.** Per-sample rank correlation coefficient of `Qwen3 base` models (left) and `posttrained` models (middle) is distributed symmetrically with mean zero. Correlation is computed within a sample `across methods` Right: Correlation coefficient and fraction of correct samples seems positively correlated. Datasets are plotted horizontally on the bottom: GPQA, Humaneval, IFEval, MATH500, MedQA, MMLU.

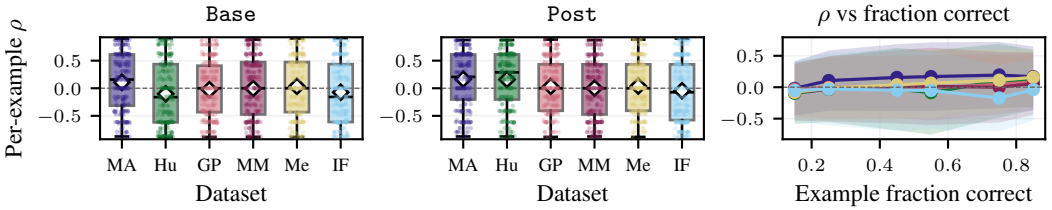


Figure 20: **Correlation coefficients are mostly symmetrically distributed around zero, the more correct a sample is, the more positive its correlation.** Per-sample rank correlation coefficient of `Qwen3 base` models (left) and `posttrained` models (middle) is distributed symmetrically with mean zero. Correlation is computed within a sample `across methods` Right: Correlation coefficient and fraction of correct samples seems positively correlated. Same as Figure 19 but excluding Beam search. Datasets are plotted horizontally on the bottom: GPQA, Humaneval, IFEval, MATH500, MedQA, MMLU.

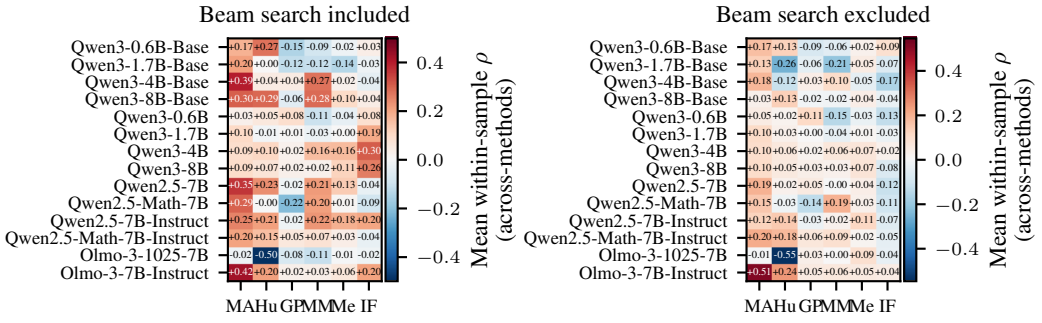


Figure 21: **Within-sample correlation coefficients are generally small with MATH500 consistently showing much larger values.** Datasets are plotted horizontally on the bottom: GPQA, Humaneval, IFEval, MATH500, MedQA, MMLU.

We show our results in Figure 19 and Figure 20 where the former include beam search and the latter do not include beam search. We do this distinction as beam search often behaves like an outlier and might influence our conclusions. We show the raw correlation coefficients in Figure 21. From the distribution of within-sample correlation we again observe a vast range of values distributed around a zero mean. Generally, beam search indeed seems to act like an outlier leading to jumps in

the correlation coefficients (even turning signs). Generally, the correlation coefficients seem to be similarly distributed as in the within-method case in Figure 17.

Focusing on the fractions of correct samples plotted against the correlation coefficients, as shown in Figure 22 and Figure 23 we find less correlation and even larger standard deviations. When excluding beam search Figure 23 curves seem to be mostly flat. Given the small sample size, we should not draw any conclusions from these results.

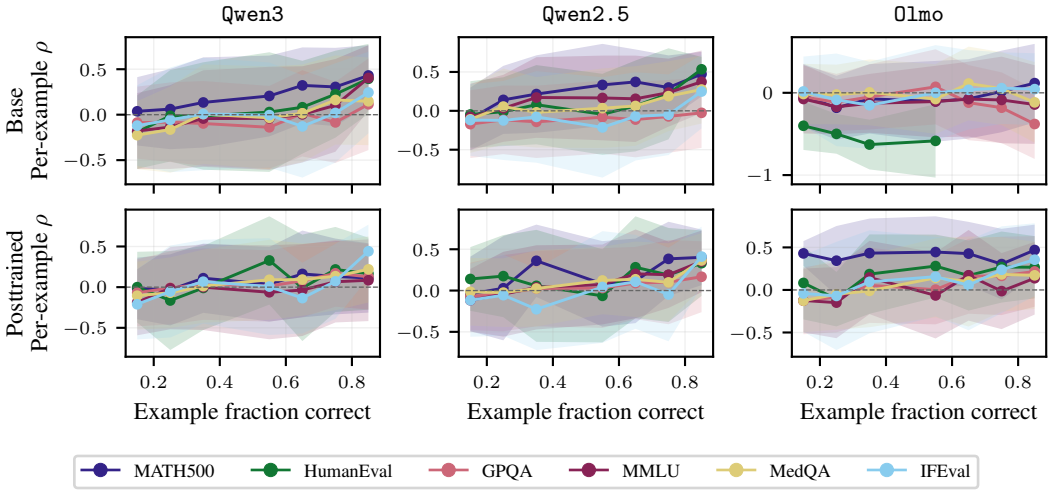


Figure 22: **The more correct a sample is under repeated sampling, the stronger its correlation coefficient.** We observe similar results across all model families (columns) and variants (rows).

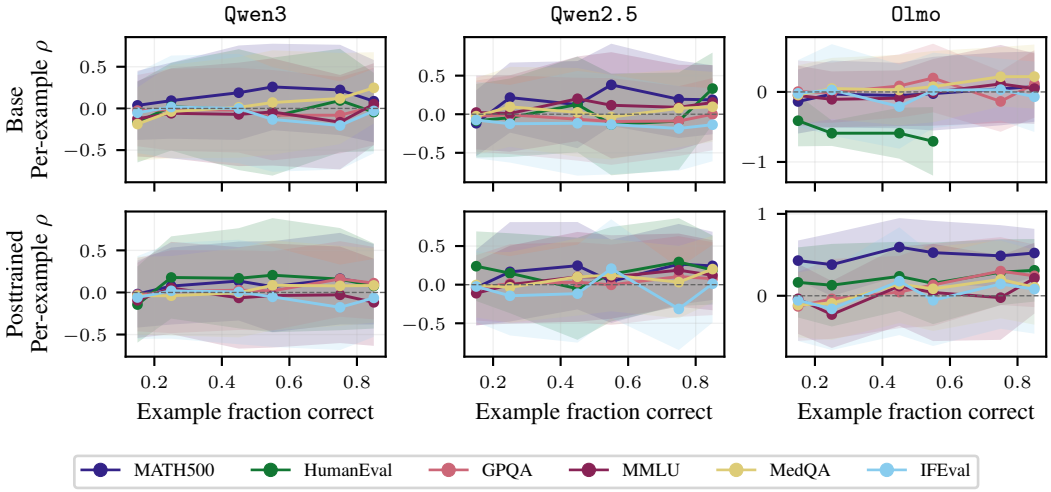


Figure 23: **The more correct a sample is under repeated sampling, the stronger its correlation coefficient.** We observe similar results across all model families (columns) and variants (rows). Same as Figure 22 but excluding beam search.

E.5 Log-Probabilities

In this section, we first analyze sequence log-probabilities in Section E.5.1. Then, we focus on token-level log-probabilities in Section E.5.2, and investigate the difference between log-probability and per-token log-probability in Section E.5.3.

E.5.1 Log-Probabilities Across Methods

In this section, we would like to understand how log-probabilities differ across local and global methods. First, in Figure 24, we are interested in global trends.

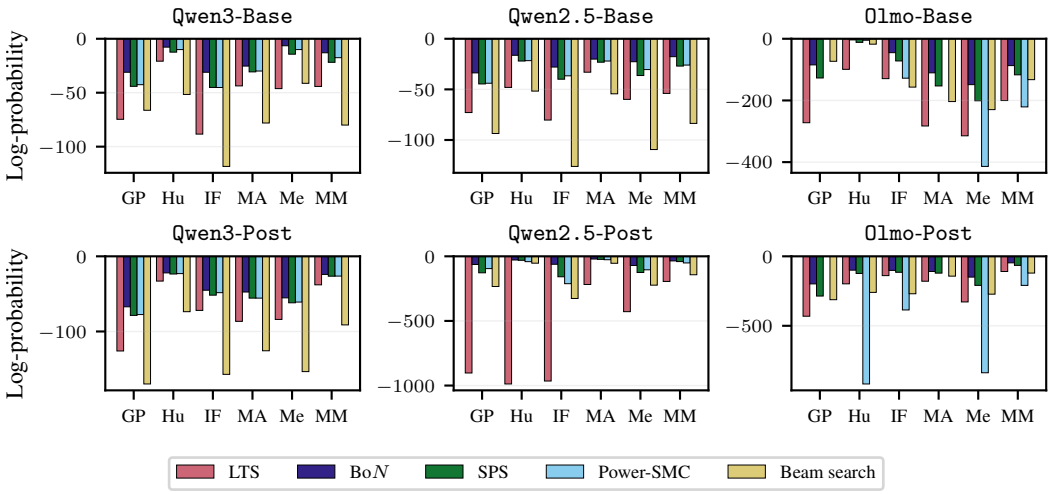


Figure 24: **Beam search often achieves smallest log-probabilities; BoN, SPS, and power-SMC have larger log-probabilities.** Log-probabilities averaged over model sizes. Typically, BoN, SPS, and power-SMC lead to smallest log-probabilities. LTS is typically larger and beam search is largest. `Olmo3` seems to be an outlier for power-SMC where log-probabilities appear very small. Datasets are plotted horizontally on the bottom of each panel: GPQA, Humaneval, IFEval, MATH500, MedQA, MMLU.

We find that there are various noticeable differences. Typically, all global methods achieve similarly large log-probabilities across models and datasets. Beam search seems to be an outlier reaching significantly lower log-probabilities, especially for `Qwen3` and `Qwen2.5`. We attribute this to the larger sequence lengths (see also Appendix E.6). LTS reaches similarly, much smaller log-probabilities. On `Olmo3`, we find power-SMC to behave like an outlier. Typically, BoN reaches the most probable sequences.

E.5.2 TOKEN-LEVEL LOG-PROBABILITIES

Next, we compare LTS, BoN, and SPS in Figure 25 and Figure 26 on a token-level. We plot per-token-log-probability in the left panels and the change of token-log-probability in the right panels for base `Qwen3-8B-Base` (top) and `Qwen3-8B` (bottom) on MATH500. Figure 26 shows the same plot for `Qwen3-1.7B`

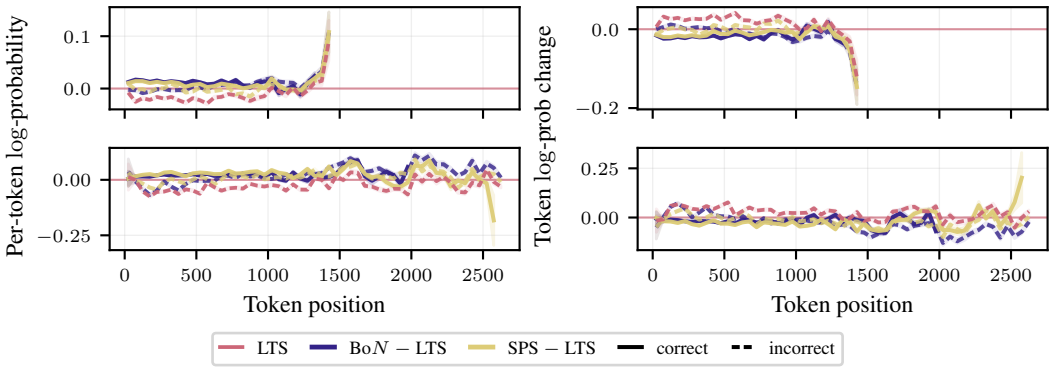


Figure 25: **Global decoding methods produce sequences of larger per-token log-probability that are smoother in `posttrained` models.** For `base` models, LTS samples are shortest (top row), SPS and BoN samples appear longer. There is no meaningful difference between log-probability and smoothness. For `posttrained` models, we observe larger per-token log-probability and smoother trajectories for global decoding methods. `Qwen3-8B-Base` (top) and `Qwen3-8B` (bottom). Details in Appendix E.5.

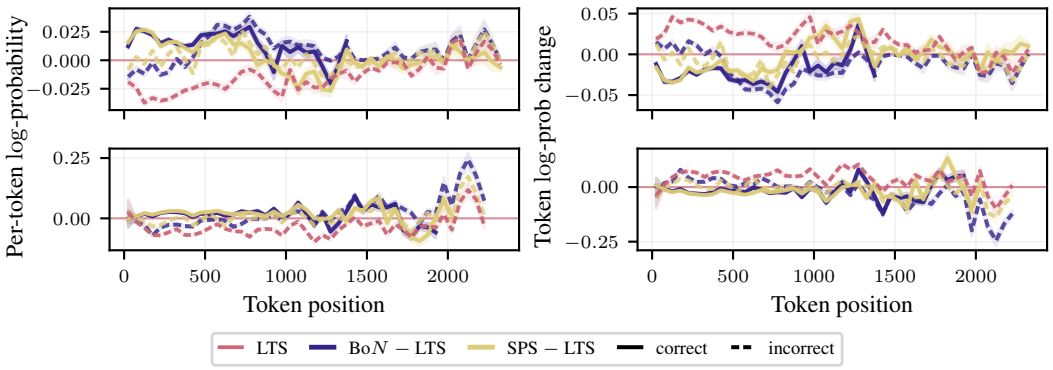


Figure 26: **Global decoding methods produce sequences of larger per-token log-probability that are smoother in `posttrained` models.** For `base` models, LTS samples are shortest (top row), SPS and BoN samples appear longer. There is no meaningful difference between log-probability and smoothness. For `posttrained` models, we observe larger per-token log-probability and smoother trajectories for global decoding methods. `Qwen3-1.7B-Base` (top) and `Qwen3-1.7B` (bottom). Details in Appendix E.5.

For `base` models, we find that samples are shorter than for `posttrained` models. Log-probabilities are smaller for incorrect sequences from LTS. There is no real difference in log-probability and smoothness comparing BoN and SPS. Generally, we find that per-token log-probabilities for global methods are larger than LTS. Similarly, global methods seem to produce sequences that are more smooth. However, we note that these effects seem to be rather small and sometimes even inconsistent.

E.5.3 Log-Probability and Per-Token Log-Probability

Here, we investigate the difference of log-probability and per-token log-probability (averaged over the sequence lengths) on a dataset level, within and across methods.

Within-dataset correlation. We investigate how our results are impacted by log-probability instead of per-token average log-probability (i.e., multiplying log-probability by the inverse sequence length). In Figure 27 we find that using per-token log-probability does not change the within-dataset correlation significantly when considering Spearman ρ. For Pearson r, many points get pushed to the corner; the off-diagonal elements seem to be consistent for ρ and r.

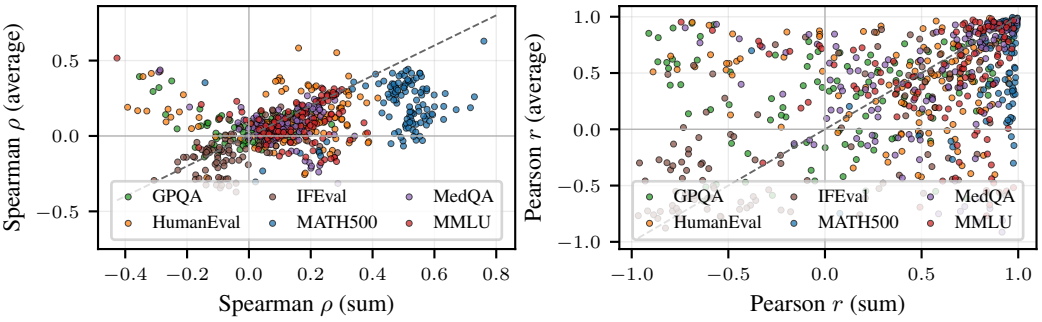


Figure 27: **With Spearman ρ there is no significant difference between log-probability and per-token log-probability while Pearson r pushes points to the corners.** Details in Appendix E.5.

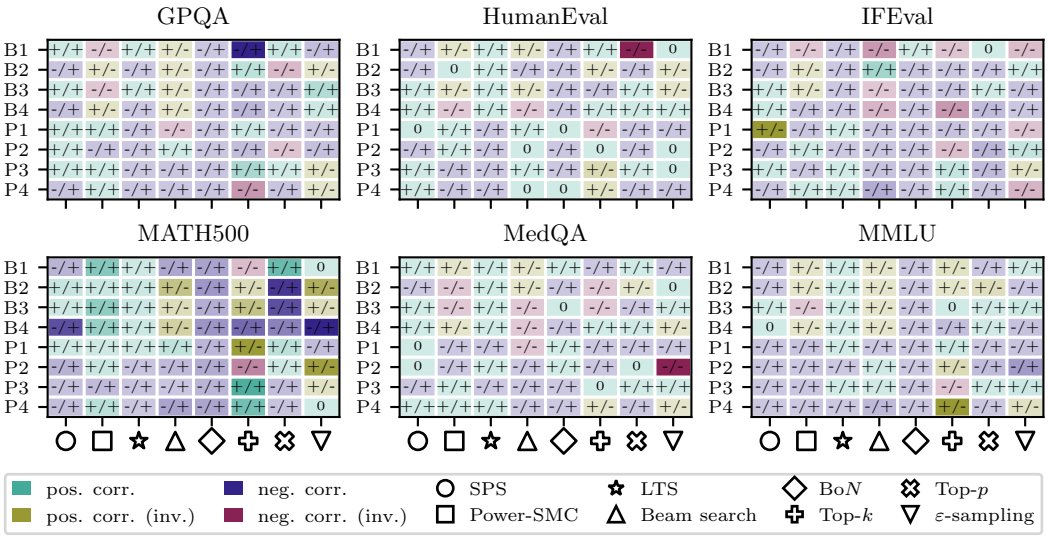


Figure 28: **Methods show either positive or negative correlation within their hyperparameter using per-token log-probability.** Correlations within methods for local and global decoding methods, models (`Qwen3` series 0.6B as ∗1, 1.7B as ∗2, 4B as ∗3, and 8B as ∗4 with base as B∗ and posttrained model as P∗), and benchmark datasets. Correlations are not consistent across datasets and models.

Within-method correlations. Considering Figure 28, Figure 29, and Figure 30, we find that mostly beam search seems to flip from negative correlation to positive correlation (inverse). Sometimes also power-SMC flipps a negative correlation. Other methods seem to not change significantly.

Across-method correlations. Across methods, it seems like many originally yellow cells (+/−) flip to red (−/−) for SPS and power-SMC aligning with the LTS implied baseline. Similarly, many of the blue cells (−/+) flip to green. For beam search, we find many yellow cells (+/−) flipping to gree cells; however, many red cells (−/−) flip to yellow (+/−). The resulting plot still does not shows a consistent direction, verifying the claims from the main text. While some of the cells flip towards more correlation, some flip to less correlation.

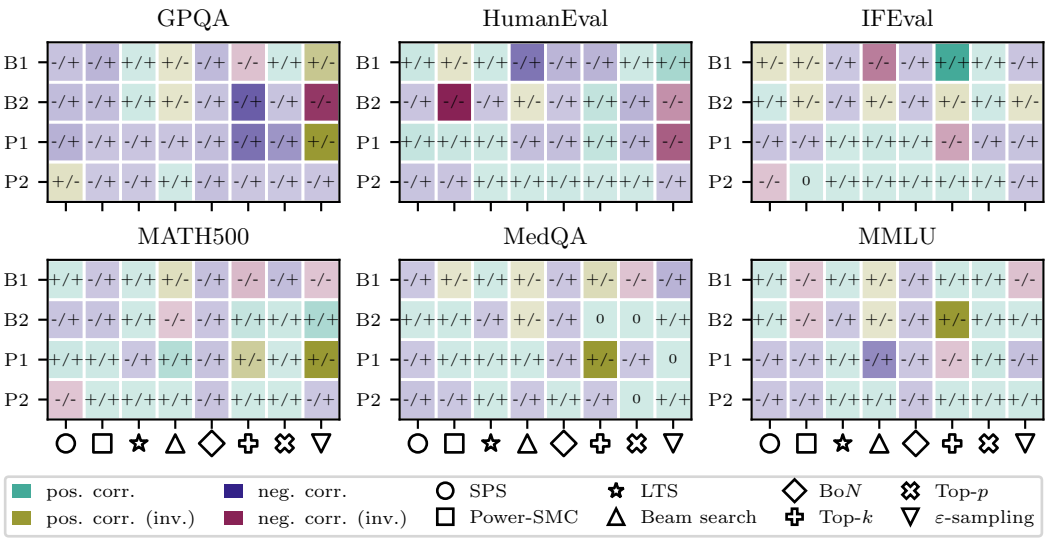


Figure 29: **Methods show either positive or negative correlation within their hyperparameter using per-token log-probability.** Correlations within methods for local and global decoding methods, models (`Qwen2.5` series 8B as ∗1, `Math` 8B as ∗2 with base as B∗ and posttrained model as P∗), and benchmark datasets. Correlations are not consistent across datasets and models.

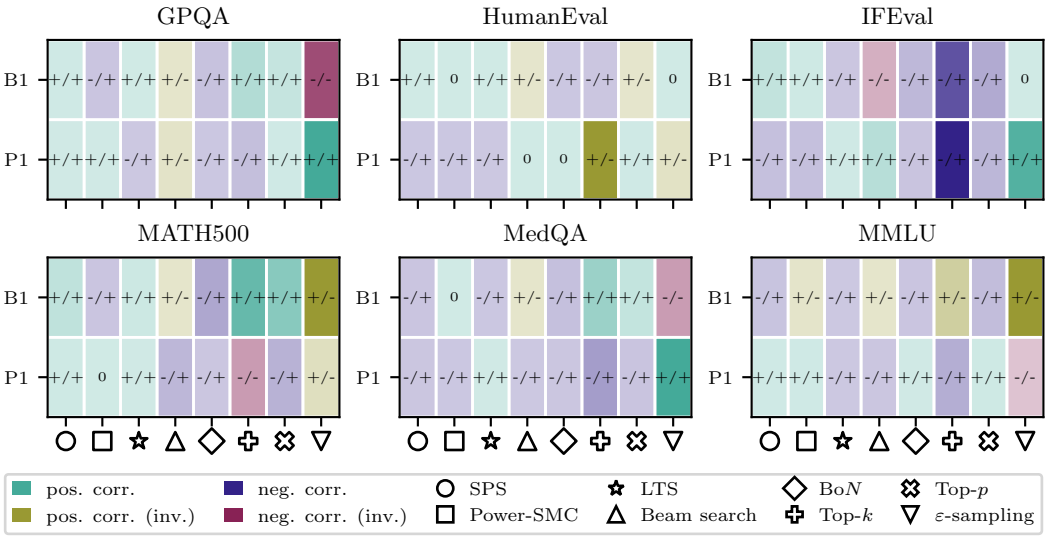


Figure 30: **Methods show either positive or negative correlation within their hyperparameter using per-token log-probability.** Correlations within methods for local and global decoding methods, models (`Olmo3` series 7B as ∗1 with base as B∗ and posttrained model as P∗), and benchmark datasets. Correlations are not consistent across datasets and models.

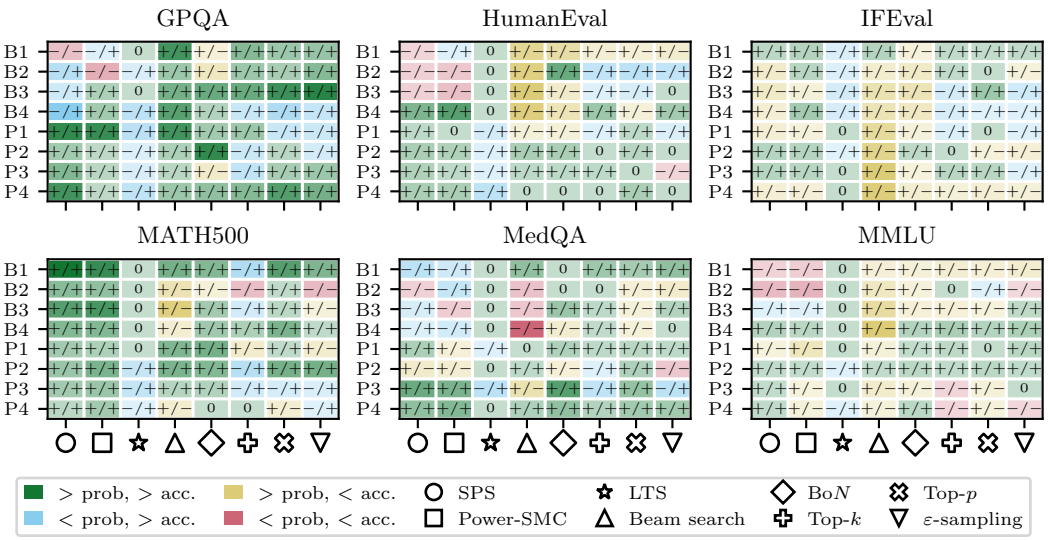


Figure 31: **Various datasets and methods show positive correlations between sequence probability and correctness using per-token log-probability.** Correlations across methods for local and global decoding methods, models (`Qwen3` series 0.6B as ∗1, 1.7B as ∗2, 4B as ∗3, and 8B as ∗4 with base as B∗ and posttrained model as P∗), and benchmark datasets. Correlations are not consistent across datasets and models.

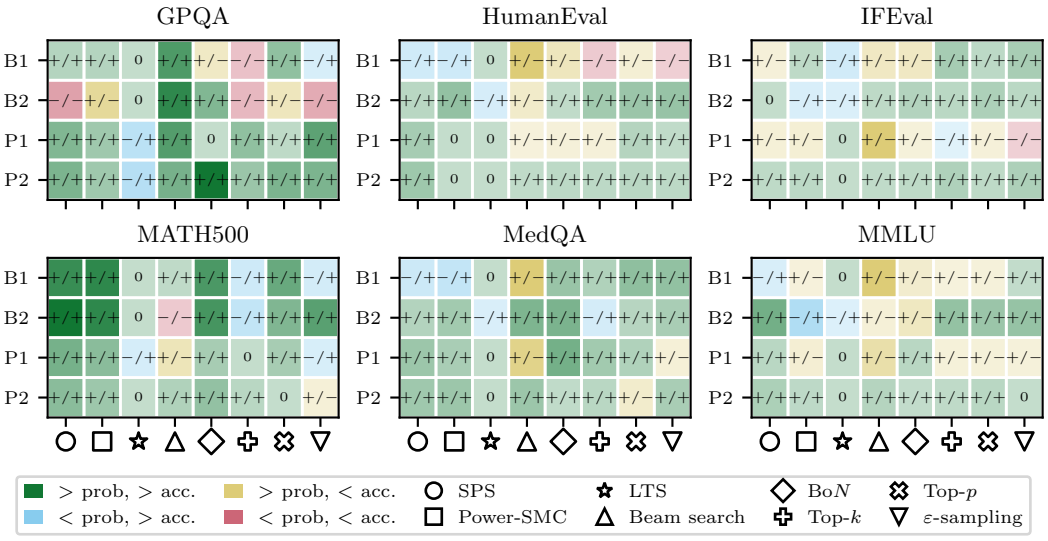


Figure 32: **Various datasets and methods show positive correlations between sequence probability and correctness using per-token log-probability.** Correlations across methods for local and global decoding methods, models (`Qwen2.5` series 8B as ∗1, `Math` 8B as ∗2 with base as B∗ and posttrained model as P∗), and benchmark datasets. Correlations are not consistent across datasets and models.

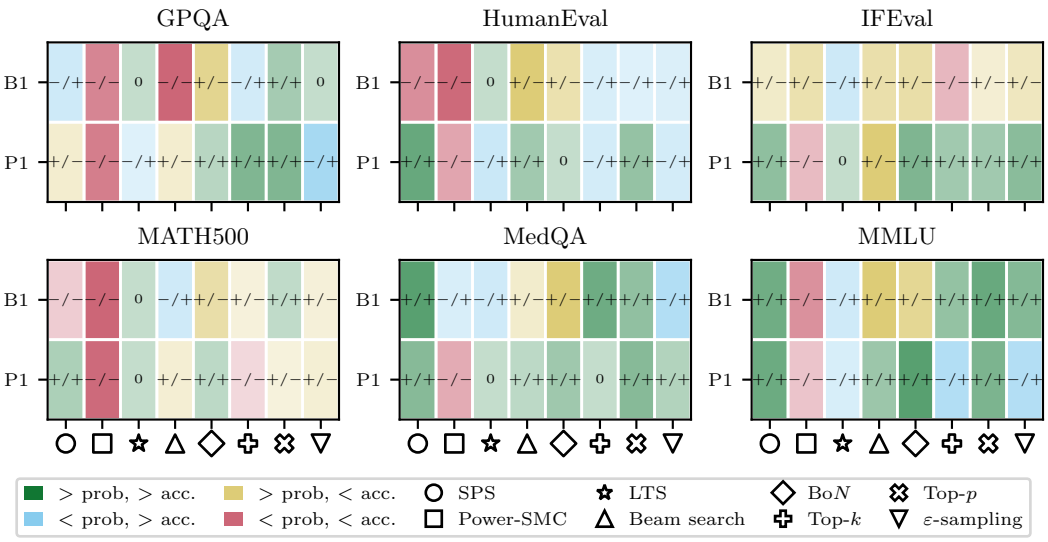


Figure 33: **Various datasets and methods show positive correlations between sequence probability and correctness using per-token log-probability.** Correlations across methods for local and global decoding methods, models(`Olmo3` series 7B as $*1$ with base as B$*$ and posttrained model as P$*$), and benchmark datasets. Correlations are not consistent across datasets and models.

E.6 SEQUENCE LENGTHS

Figure 34 and Figure 35 show `Qwen3-8B base` and `posttrained` models, respectively. We find that the sequence lenghts are very similar across methods. Beam search appears to be the only exception finding very long sequences.

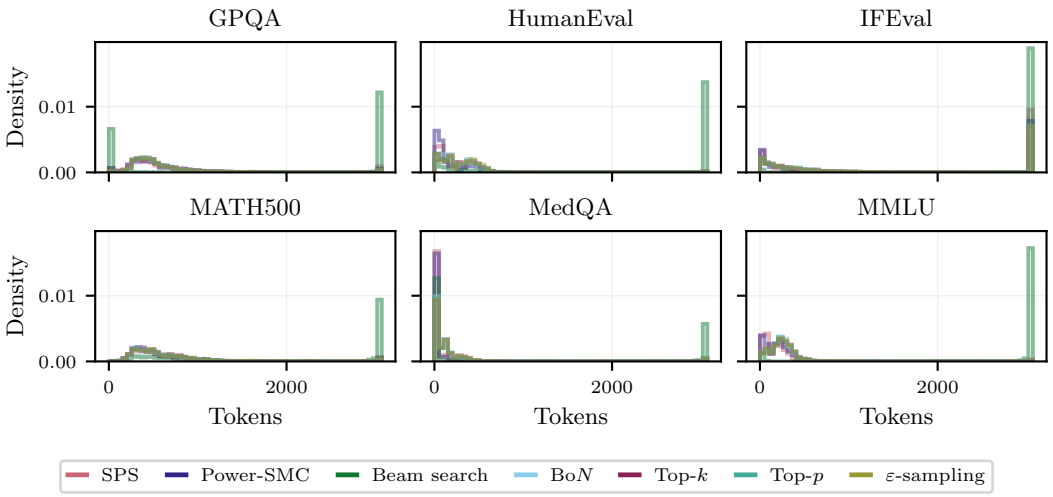


Figure 34: **Sequence lengths overlap across methods, beam search is the only exception.** Data for `Qwen3-8B-Base`. Details in Appendix E.5.

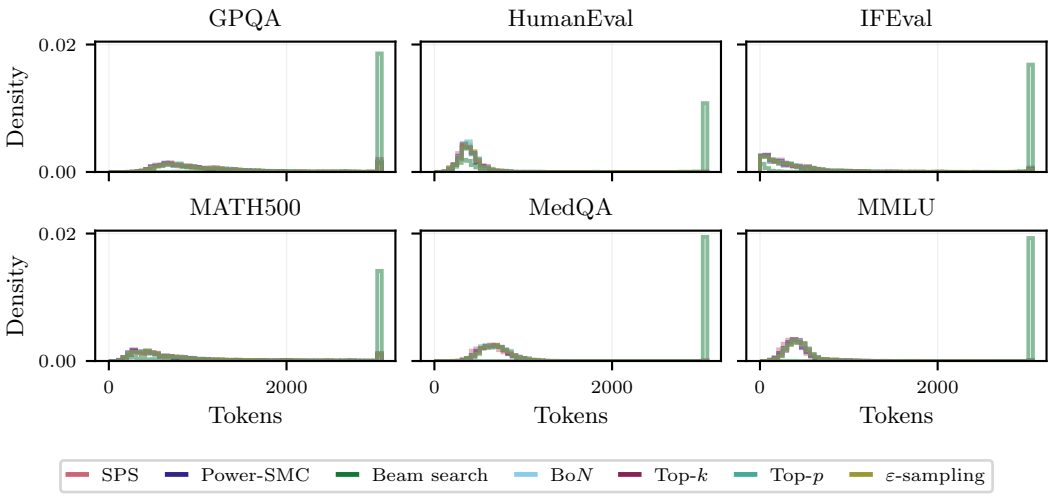


Figure 35: **Sequence lengths overlap across methods, beam search is the only exception.** Data for `Qwen3-8B`. Details in Appendix E.5.

E.7 Mode and Power Distribution

In Section 2, we found that the power-distribution can be understood as a soft mode-seeking objective. Here, we compare both distributions using BoN (which returns the mode for $N \to \infty$) for the mode and SPS and power-SMC for the power distribution.

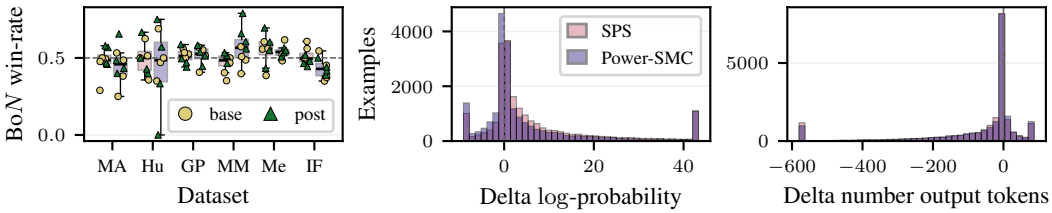


Figure 36: Left: **Comparing mode and power-distribution we find no consistent differences in accuracy.** Middle, right: **BoN typically leads to shorter samples that are of smaller probability.** Datasets are plotted horizontally on the bottom of each panel: GPQA, Humaneval, IFEval, MATH500, MedQA, MMLU. `Qwen3` model family.

Comparing correctness across BoN and power-distribution in Figure 36 (leftmost panel), we find no consistent trend across datasets and models (and variants). In the panel, we plot the win-rate of BoN against SPS and power-SMC across datasets and the `Qwen3` series. Investigating the samples in detail, we find that samples from BoN seem to be typically of smaller probability (Figure 36, middle) and shorter, two properties that influence each other.

E.8 SELF-DISTILLATION

Hinton et al. (2015) distill an ensemble of neural networks in a single neural network. Zhang et al. (2019) propose self-distillation for convolutional neural networks where knowledge is distilled within a network itself. In the context of LLMs, the self-taught reasoner (Zelikman et al., 2022) consists of a bootstrapping method that lets LLMs generate rationales for prompts which are then used for finetuning. Chen et al. (2024a) frames reasoning as sampling from a latent distribution optimizing it via sequence likelihood as self-rewarding signal.

Here, we train self-distillation on BoN, SPS, and power-SMC samples using the first half of samples considered. We test the model consistently on the second half of samples.

For fine-tuning we choose a LoRA with $r = 16$, $\alpha = 32$ and dropout of 0.05 for 1 epoch with a learning rate of $2 \cdot 10^{-5}$ and a warmup ratio of 0.03.

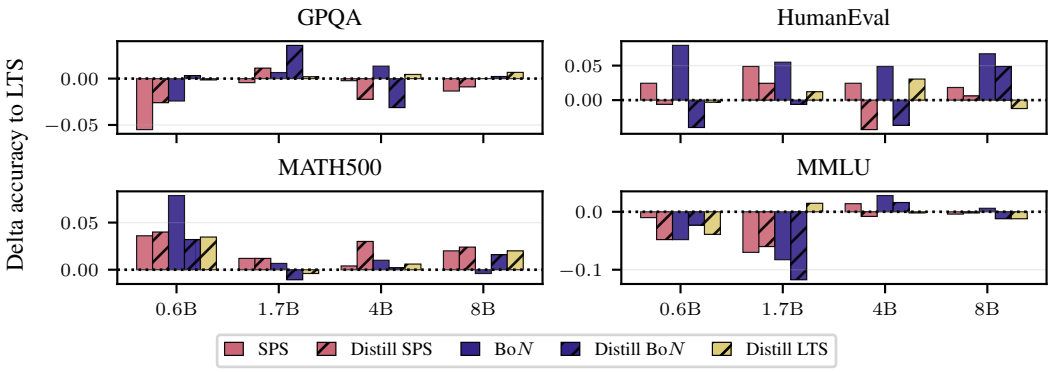


Figure 37: **Self-distillation shows diverse results across methods.** On MATH500 power self-distillation seems to often improve results. Else, results are mixed. `Qwen3 base` models.

In Figure 37 we find that self-distillation seems to work well on samples where a correlation between log-probability and correctness can be found (i.e., MATH500). Else we observe mixed results. Further, self-distillation on samples from the power distribution works best where those samples are outperforming the baseline LTS (MATH500). Else, simple distillation on LTS samples often outperforms power samples.

E.9 QWEN3 THINKING

In this section, we enable thinking for the Qwen3 model series but keep the maximum number of tokens as before.

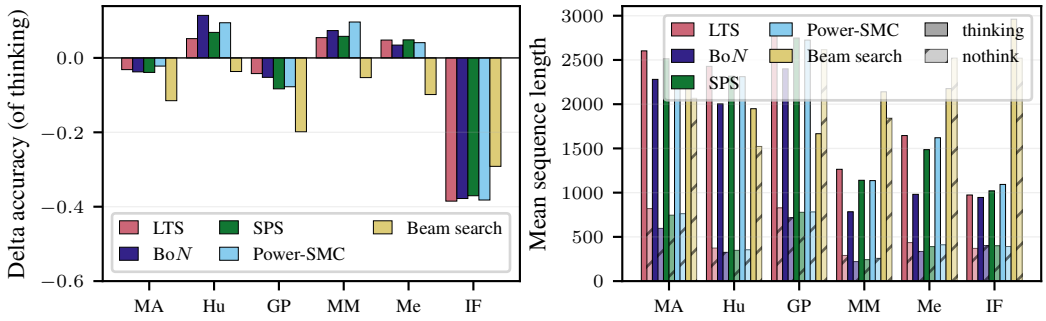


Figure 38: Left: **Qwen3 thinking mostly improves on Humaneval, MMLU, and MedQA; the strongest decline can be observed on IFEval.** Across methods, beam search often performs worst. Right: **Thinking models often reach the token limit.** The mean sequence length for thinking models is much higher as compared to non-thinking models.

In Figure 38 (left) we find that Qwen3 thinking models often degrade the performance (3 out of 6 datasets) across methods (with beam search acting as an exception). We explain this mostly by the fact that they reach the token limit (due to producing many tokens inside the <think>-tags) and do not come to an answer (Figure 38, right). As the token limit is dictated by our computational constraints, we focus in the main text solely on the non-thinking version. It seems interesting to rerun our experiments with thinking models in future work.