QUANTIFYING REASONING PERFORMANCE WITH FORMATTING CONSTRAINTS

Kian Kyars
Independent
Edmonton
December 2025
kiankyars@gmail.com

ABSTRACT

In this paper, I seek to answer the question of whether forcing a model to adhere to complex non-functional formatting rules degrades its ability to reason, by proxy of performance on a PhD-level reasoning benchmark. My objective is to provide actionable insights on the extent to which there exists a formatting tax on reasoning capabilities in AI agents, which will be useful for the engineering community. I use one of the current SoTA reasoning models, Claude Opus 4.5, on the Diamond GPQA benchmark, which is shown on the system card of all Frontier Lab models, to test how different reasoning constraints affect benchmark performance.

***K*eywords** Reasoning · Formatting · GPQA · CoT

1 Introduction

Although it has been one year since the mainstream arrival of reasoning models, many aspects of their behavior are only understood weakly, and robust experimentation can strengthen our collective understanding. A better understanding on how prompting affects reasoning can help those using thinking models in their day-to-day workflow to better take advantage of them.

2 Related Work

Factory [3] showed that context compression hurts agentic behavior; we focus on *output* formatting. GPQA Diamond is used in frontier model cards [1] as a high bar for expert-level reasoning.

3 Methodology

3.1 Reasoning Models

In this study, I test with Claude Opus 4.5 (claude-opus-4-5-20251101), using the same parameters as the official Opus 4.5 GPQA model card results, which are located in Appendix B.

3.2 Benchmark

The Graduate-Level Google-Proof Q&A benchmark (GPQA) is a set of very challenging multiple-choice science questions. The GPQA Diamond subset of 198 questions are described by the developers of the test as the “highest quality subset which includes only questions where both experts answer correctly and the majority of non-experts answer incorrectly” [2]. Furthermore, if an "expert validator answers incorrectly . . . they [must] describe clearly the mistake or their understanding of the question writer’s explanation".

3.3 Formatting Constraints

Each condition uses the same task and answer rule: the last line must be `solution: X` with $X \in \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}\}$. I add the following constraints on the *reasoning* preceding the solution:

1. **Baseline (Prompt 0):** Identical to harness used in Opus 4.5 model card.
2. **Strict JSON (Prompt 1):** The model must output valid JSON only, containing exactly five keys: `initial_intuition`, `step_by_step_logic`, `potential_counterarguments`, `confidence_score_0_to_1`, and `solution`.
3. **Structural Rigidity (Prompt 2):** Reasoning must consist of exactly three bullet points, each no longer than 20 words, and must not use the words "because" or "therefore".
4. **Python Code (Prompt 3):** The model must write its reasoning in Python.
5. **Oulipo Constraint (Prompt 4):** The letter 'e' cannot appear anywhere in the reasoning chain, inspired by the Oulipo literary movement.
6. **Restricted Vocabulary (Prompt 5):** Reasoning cannot use 16 specific high-norm English tokens identified from GPT-oss embeddings: accordingly, code, ocode, The, settings, Moreover, description, Let's, This, core, utilizes, revolves, Here's, possibly, logic, thereby.

4 Experimental Setup

I evaluate each formatting constraint on the full GPQA Diamond dataset (198 questions) with 5 repetitions per question-constraint pair, resulting in 990 total evaluations per condition. Questions are presented with randomly shuffled answer choices to prevent position bias. All experiments use Claude Opus 4.5 with identical parameters (see Appendix B) to ensure fair comparison across conditions.

Accuracy is calculated as the fraction of correct answers per condition, aggregated across all questions and repetitions.

5 Results

5.1 Accuracy by Formatting Constraint

Table 1 presents accuracy results.

Table 1: Accuracy and Token Usage by Prompt Type

Prompt Type	Accuracy	Avg Input Tokens	Avg Output Tokens
Baseline	0.872 ± 0.017	311	12358
Strict JSON	0.855 ± 0.014	368	12510
Structural Rigidity	0.858 ± 0.007	353	9749

5.2 Token Usage Analysis

Figure 1 in Appendix A shows token usage patterns across conditions.

5.3 Error Analysis

6 Conclusion

Your conclusion here

Acknowledgments

I thought of this idea after reading an article by Factory on the importance of context formatting for agentic performance [3].

A Figures

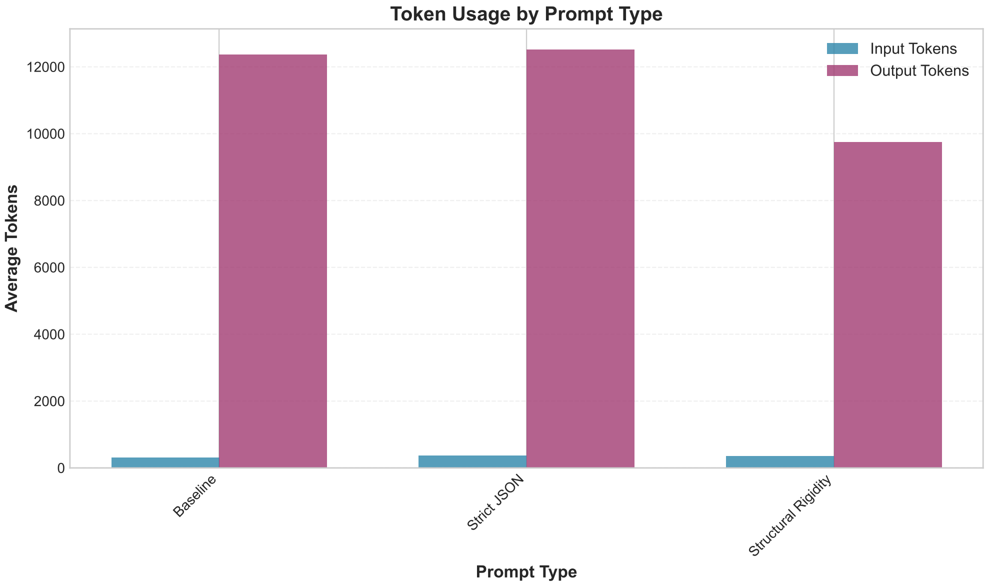

Figure 1: Token usage by formatting constraint

B Model and API Parameters

I use the Messages API with: `model=claude-opus-4-5-20251101`; `thinking={type: enabled, budget_tokens: 64000}`; `output_config.effort=high`; `max_tokens=64000`. Betas: `interleaved-thinking-2025-05-14`, `effort-2025-11-24`. This matches the setup used for GPQA in the Opus 4.5 system card [1].

References

[1] Anthropic. System card: Claude opus 4.5, 2025. `https://assets.anthropic.com/m/64823ba7485345a7/Claude-Opus-4-5-System-Card.pdf`.

[2] 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. arXiv:2311.12022, 2023. `https://arxiv.org/abs/2311.12022`.

[3] Factory Research. Evaluating context compression for ai agents, 2025. `https://factory.ai/news/evaluating-compression`.