FORMATTING TAX: HOW CONSTRAINTS AFFECT REASONING

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. My findings show that among the reasoning constraints I subject the model to, all degrade accuracy on the benchmark similarly, and that unconstrained Reasoning yields the best output

***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 [**?**] showed that context compression hurts agentic behavior; I focus on *output* formatting. GPQA Diamond is used in frontier model cards [**?**] 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 **??**.

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” [**?**]. 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 $X \in \{\text{A}, \text{B}, \text{C}, \text{D}\}$. I add the following constraints on the *reasoning*:

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, and 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 **??**) 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 **??** presents accuracy results.

5.2 Token Usage Analysis

Figure **??** in Appendix **??** shows token usage patterns across conditions. Structural Rigidity yields the feweest output tokens, consistent with the 20-word-per-bullet and three-bullet limit.

5.3 Error Analysis

6 Limitations

The study uses a single model (Claude Opus 4.5) and one benchmark (GPQA Diamond). Generalization to other reasoning models and benchmarks is unknown.

7 Conclusion

What I conclude from this study is that unconstrained formatting gives higher accuracy on the reasoning benchmark tested. There does not seem to be a specific format constraint which lobotomizes the model more than others.

Acknowledgments

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

Table 2: Accuracy by subdomain and prompt type.

Subdomain	Baseline	Strict JSON	Structural Rigidity	Python	Oulipo	Banned Words
Astrophysics	1.000	0.969	1.000	0.985	1.000	0.985
Chemistry (general)	0.860	0.810	0.790	0.820	0.790	0.800
Condensed Matter Physics	1.000	1.000	1.000	1.000	1.000	1.000
Electromagnetism and Photonics	0.867	0.867	0.833	0.867	0.833	0.900
Genetics	0.750	0.750	0.750	0.750	0.750	0.750
High-energy particle physics	1.000	1.000	1.000	1.000	1.000	1.000
Inorganic Chemistry	1.000	1.000	1.000	1.000	1.000	1.000
Molecular Biology	0.800	0.800	0.787	0.773	0.773	0.773
Optics and Acoustics	1.000	1.000	1.000	1.000	1.000	1.000
Organic Chemistry	0.792	0.781	0.792	0.786	0.819	0.794
Physics (general)	0.926	0.895	0.905	0.895	0.895	0.895
Quantum Mechanics	0.984	0.960	0.960	0.976	0.952	0.968
Relativistic Mechanics	0.857	0.857	0.857	0.857	0.857	0.857

A Figures

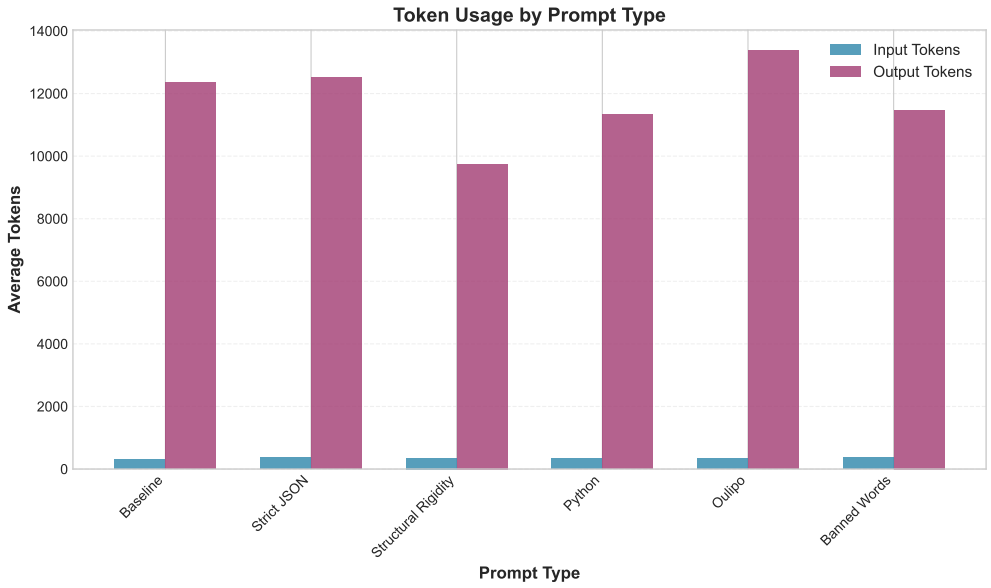

Figure 1: Token usage by formatting constraint

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 [**?**].

C Accuracy by Subdomain and Prompt

D Prompts and Reproducibility

Code and prompts are available at `https://github.com/kiankyars/gpqa`. The six prompt variants (Baseline, Strict JSON, Structural Rigidity, Python Code, Oulipo, Restricted Vocabulary) are defined in `main.py`.

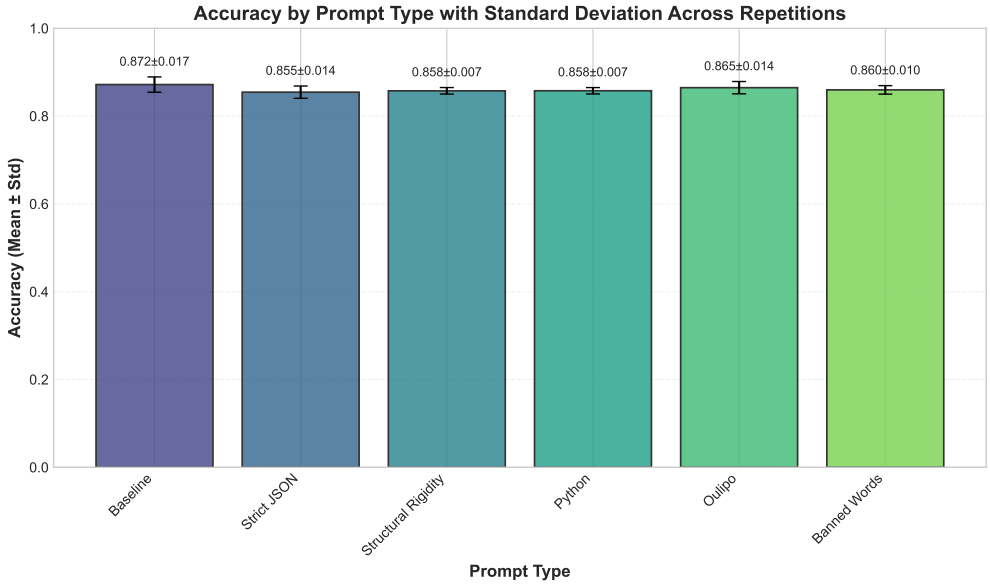

Figure 2: Accuracy with 95% intervals by formatting constraint

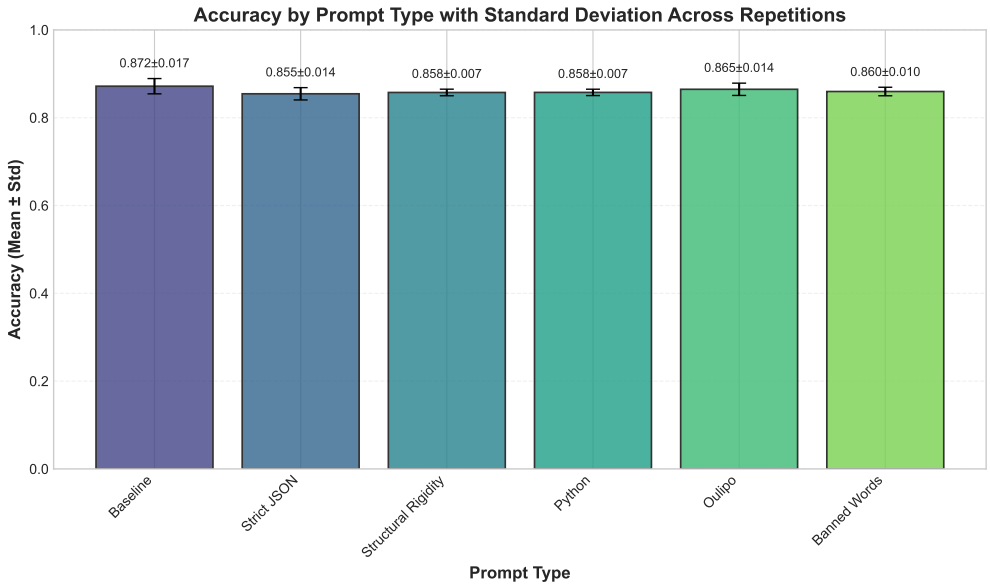

Figure 3: Accuracy with 95% intervals by formatting constraint