Beneath the Surface of Chains-of-Thought: A Mechanistic Interpretation of Reasoning Operations in LLMs

Anonymous ACL submission

Abstract

Reasoning in large language models evolves through a sequence of diverse cognitive operations such as problem formulation, goal division, and deduction. While these sequence of reasoning operations are explicitly expressed and distinguished in text, little is known about how they are geometrically organized in representation spaces. To this end, we investigate whether textually distinctive reasoning operations exhibit corresponding representational geometry. We find that reasoning operations are separable in hidden space, with peak separability in middle layers. To understand what enables this separability, we analyze how reasoning operation-relevant representations emerge across layers. We observe that even identical texts are encoded toward representations of corresponding reasoning-operation depending on the reasoning operation of their containing chunk. To understand what triggers these reasoning operations, we conduct causal intervention via attention masking of preceding context. We demonstrate that preceding context significantly influences reasoning representation, revealing that reasoning operations are not locally self-contained but triggered through information propagation from prior context. Consequently, our work demonstrates that language models maintain representational correspondence between linguistic reasoning expressions and their internal geometric structures.

1 Introduction

Recent reasoning-oriented large language models have shown strong performance on multi-step problem-solving tasks (Xu et al., 2025a). A central mechanism behind this progress is chain-of-thought reasoning (Wei et al., 2022), where models generate intermediate rationales to derive answers. These rationales make the problem-solving process partially observable. For example, the models may decompose the problem, retrieve relevant information, perform intermediate computations, apply rules, and verify partial conclusions. As a result, understanding the reasoning process itself beyond the correctness of answer has emerged as an essential challenge for interpreting and improving LLM behavior.

However, while reasoning traces are observable in text, they provide only a superficial descriptions of the model's underlying computation. Recent latent and continuous reasoning methods suggest that reasoning is not fully expressed through discrete natural-language tokens but can also be supported by hidden-state dynamics (Hao et al., 2024; Shen et al., 2025; Xu et al., 2025b). Complementary representation-level studies further show that reasoning leaves structured signatures in hidden space, where correctness-related information is encoded during reasoning (Zhang et al., 2025a) and chain-of-thought generation traces structured trajectories through latent space (Sun et al., 2026). These findings indicate that reasoning is accompanied by systematic changes in internal representations. However, how the macroscopic flow of distinct reasoning operations within a trace is organized in these representations remains an open question.

In this study, we aim to characterize how textually explicit reasoning operations are represented in the hidden spaces of large language models. We view reasoning as a sequence of distinct operations and investigate whether these operations are reflected in the internal representational geometry of reasoning LLMs. To this end, we investigate the following key questions: (1) Do reasoning operations that are explicitly expressed and distinguished in text form geometric structures in representation spaces? (2) How do the geometric structures of reasoning operation emerge? (3) What contextual factors give rise to these geometric structures?

To answer these questions, we categorize reasoning operations using Polya's problem-solving

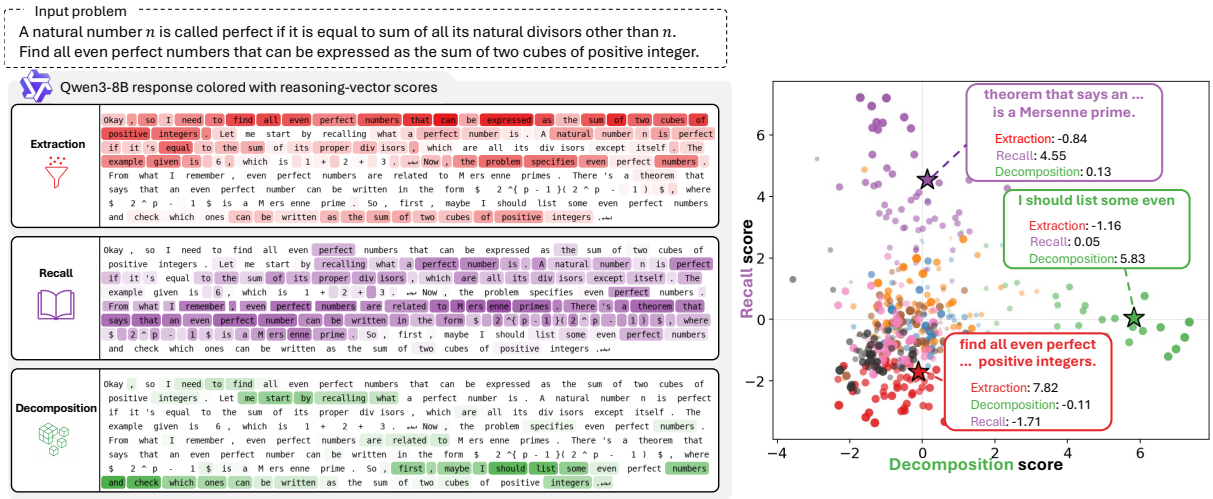


Figure 1: **Overview of reasoning-vector analysis.** Given an input problem and the model-generated reasoning trace, we probe latent representations with reasoning vectors corresponding to different reasoning operations. The left panel shows token-level operation scores over the generated text, while the right panel projects span representations onto *Decomposition* and *Recall* reasoning-vector axes. Regions with strong reasoning operation-specific signals appear along the corresponding directions.

framework from *How to Solve It* (Polya, 1945), which provides a coarse-grained taxonomy of stages. We apply this taxonomy to generated reasoning traces and analyze the corresponding hidden representations on mathematical reasoning datasets, including DAPO-17K (Yu et al., 2026) and TheoremQA (Chen et al., 2023). We further examine multiple reasoning-oriented LLMs from the Qwen and Gemma families to assess whether operation-level structures are consistently observed across model families. Our key findings are summarized as follows:

- **Representational separability of reasoning operations (§3.2):** Reasoning operations form separable clusters in hidden representations, with peak separability emerging in middle layers. This indicates that the functional distinctions visible in text are preserved as structured geometry in latent space.

- **The geometric structure evolves across layers (§3.3):** These structures emerge progressively across layers, with even representations identical tokens vary depending on their containing span. This indicates that separability arises from contextual processing rather than operation-specific vocabularies.

- **Preceding contexts triggers the subsequent reasoning operations (§3.4):** Causal intervention via attention masking shows that reasoning operations are not locally self-contained within their text spans. Instead, they are triggered and reinforced through information propagation from preceding context.

Our findings suggest that language models maintain a representational correspondence between explicitly expressed reasoning operations in text and their internal geometric organization.

2 Preliminary

Reasoning LLMs. Given an input prompt x, a reasoning LLM generates an intermediate reasoning trace $r = (t_1, \ldots, t_N)$ before producing the final answer y:

$$p(r, y \mid x) = \prod_{i=1}^{N} p(t_i \mid x, t_{<i}) \cdot p(y \mid x, r). \quad (1)$$

For each token position i, we extract hidden representations $\mathbf{h}_i^{(\ell)} \in \mathbb{R}^d$ from layer ℓ of the model. Our study investigates whether and how the geometric organization of these representations reflects the functional structure of the reasoning process expressed in text.

Operational reasoning type taxonomy. We view the reasoning trace r not as a homogeneous token sequence but as a sequence of *reasoning operation chunks*, $r = (c_1, \ldots, c_K)$, where each chunk c_k corresponds to a contiguous span of tokens that serves a specific functional role in the problem-solving process. To systematically categorize these operations, we define a hierarchical taxonomy of

Reasoning Stage	Reasoning Type	Example
Understanding the problem	Extraction Direct mapping	The problem says that **two numbers add up to 10.** x is greater than 3 → $x > 3$
Planning the solution	Decomposition	Get the area of Heptagon → **Divide it into triangles** → **Solve each triangle** → Combine results
Carrying out the plan	Recall Deduction Algebraic manipulation Arithmetic computation	Area of a circle is needed → **Recall** $A = \pi r^2$ → apply to problem x is greater than 3 and x is an integer → $x \geq 4$ $x + y = 10 \rightarrow y = 10 - x$ $10 - 3 = 7$
Looking back and final answer	Final answer	Therefore, the values are $x = 3$ **and** $y = 7$.

Table 1: **Overview of main reasoning operations for our analyses.** These eight frequent operation types are used as span labels in our representation analysis.

reasoning operations for generated reasoning traces based on the four-stage structure (Polya, 1945). The taxonomy is intended to capture the functional role of each reasoning step in the solution process, rather than its surface wording alone. For example, a step may extract given information from the problem, translate a natural-language statement into a symbolic form, decompose the problem into subproblems, recall a relevant formula or theorem, derive a conclusion, manipulate an equation, perform arithmetic computation, or present the final answer.

The full taxonomy includes a broader set of operation types and subtypes. In the main analysis, we focus on eight recurring operation types that appear frequently across the generated traces and provide sufficient examples for representation-level analysis. The eight main operation types cover different parts of four Polya's problem-solving process. In the **understanding stage**, Extraction identifies information explicitly given in the problem, while Direct mapping converts natural-language statements into formal expressions. In the **planning stage**, Decomposition captures the act of breaking a complex problem into smaller subproblems. In the **execution stage**, Recall retrieves relevant formulas, definitions, or rules; Deduction derives conclusions from known premises; Algebraic manipulation transforms symbolic expressions; and Arithmetic computation performs numerical calculations. Finally, Final answer closes the reasoning process by presenting the derived result in the required format. The examples of these main eight reasoning taxonomy is shown in table 1. The remaining types are used to organize the taxonomy and are described in Appendix E.

We emphasize that this taxonomy is an operational analysis framework, not a definitive cognitive model of LLM reasoning. Labels are assigned based on the textually explicit function of each step. Therefore, the taxonomy allows us to examine whether different explicit reasoning operations are associated with distinguishable hidden-state structures, without assuming that the labels exhaustively describe the model's full latent computation.

3 Geometric Structure of Reasoning Operations in Reasoning LLMs

3.1 Experimental Setup

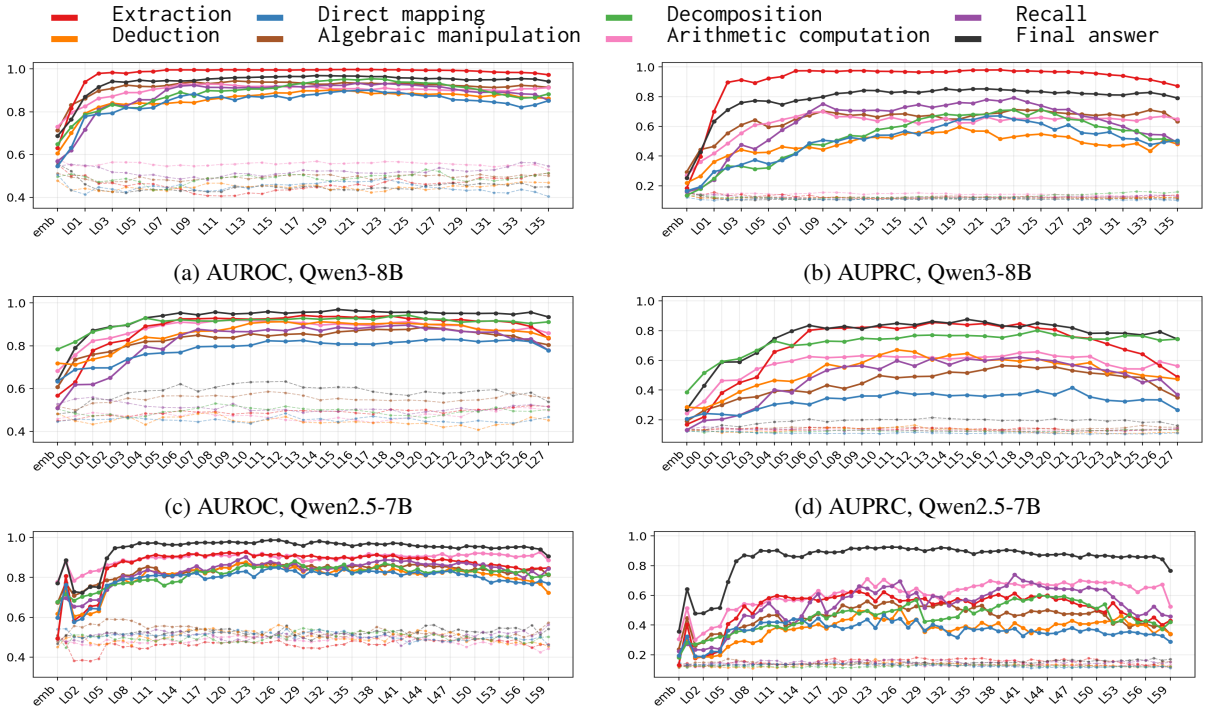
Tasks and data. We use two reasoning datasets, DAPO-Math-17K (Yu et al., 2026) and TheoremQA (Chen et al., 2023). DAPO-Math-17K consists of mathematical reasoning problems, while TheoremQA contains theorem-driven questions that require applying domain knowledge to solve problems. Using both datasets allows us to examine reasoning operations in computation-heavy mathematical reasoning as well as more theorem-based reasoning settings.

Models. We analyze three reasoning LLMs: `Qwen2.5-7B` (Qwen et al., 2025), `Qwen3-8B` (Yang et al., 2025), and `Gemma4-31B`. This validates the generalizability of our observations beyond a specific model family.

Reasoning Taxonomy. We use the eight main reasoning operation types introduced in the Preliminary section: Extraction, Direct mapping, Decomposition, Recall, Deduction, Algebraic manipulation, Arithmetic computation, and Final-answer.

3.2 Representational Separability of Reasoning Operations

Reasoning Operation Labeling. For each model and dataset, we generate reasoning traces and retain 500 – 700 correct responses. Specifically, each reasoning trace r is then segmented into a sequence of non-overlapping reasoning operation



(a) AUROC, Qwen3-8B (b) AUPRC, Qwen3-8B

(c) AUROC, Qwen2.5-7B (d) AUPRC, Qwen2.5-7B

(e) AUROC, Gemma4-31B (f) AUPRC, Gemma4-31B

Figure 2: **Reasoning operations are separable in the hidden representation space.** AUROC and AUPRC of one-vs-rest operation classifiers are shown across layers for `Qwen3-8B`, `Qwen2.5-7B`, and `Gemma4-31B` using middle-token span representations. Solid lines denote true reasoning-operation labels, while dashed lines denote random-label and random-position baselines. Across models, operation scores remain substantially above the baselines and peak in middle layers.

spans $\{s_1, s_2, \ldots, s_K\}$, where each span $s_i = (t_i^{\text{start}}, t_i^{\text{end}}, y_i)$ denotes a contiguous token range $[t_i^{\text{start}}, t_i^{\text{end}}]$ annotated with a reasoning operation label. We annotate each trace into reasoning spans using GPT-5, assigning each span one of the eight main reasoning operation labels.

Experiment. We evaluate whether reasoning operations form separable structures in hidden representation space. For each annotated span $s_i = (t_i^{\text{start}}, t_i^{\text{end}}, y_i)$, we extract hidden representations from each layer and representation type. In the main results, we use the middle-token representation of each span; alternative pooling strategies, including first token, last token, and mean pooling, are reported in Appendix D. Let $x_i^{(l,m)}$ denote the span representation at layer l and representation type m.

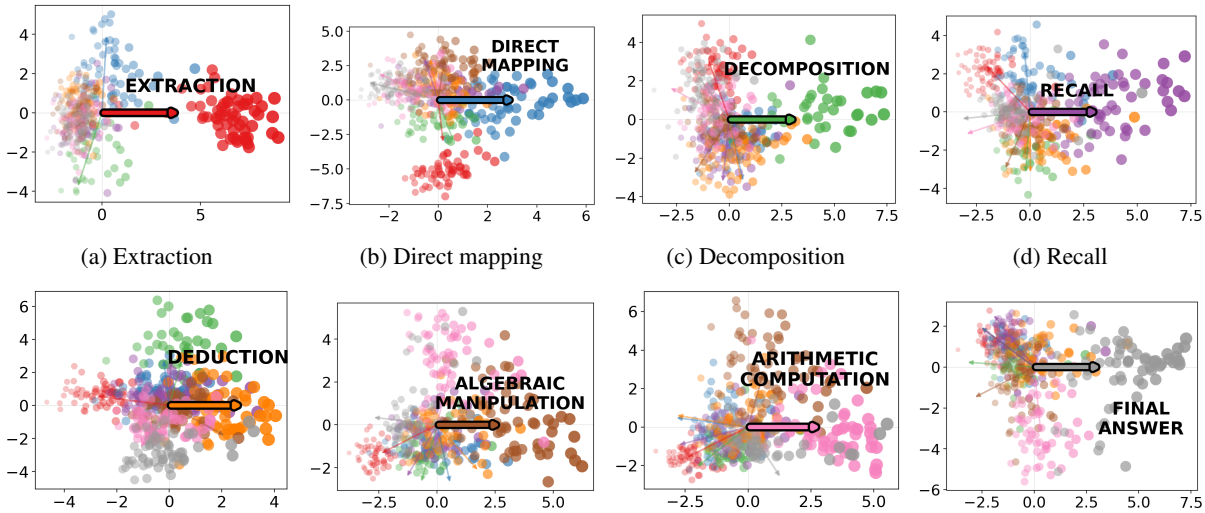
For each layer and representation type, we split spans into train and test sets. Using only the train split, we L_2-normalize the span representations, apply PCA to 128 dimensions, and then fit supervised LDA using the reasoning-operation labels. The learned PCA and LDA projections are then applied to the held-out test split. Since we use $C = 8$ operation classes, the LDA space has at most $C - 1 = 7$ dimensions. Let $z_i^{(l,m)}$ denote the projected representation of $x_i^{(l,m)}$.

For each operation c, we compute a one-vs-rest operation direction in the projected space: $d_c^{(l,m)} = \frac{\mu_c^{(l,m)} - \mu_{\neg c}^{(l,m)}}{\left\| \mu_c^{(l,m)} - \mu_{\neg c}^{(l,m)} \right\|_2}$, where $\mu_c^{(l,m)}$ is the centroid of train spans labeled as c, and $\mu_{\neg c}^{(l,m)}$ is the centroid of all other train spans. For each test span i, we compute its alignment with operation c as $a(i, c) = \left(z_i^{(l,m)}\right)^\top d_c^{(l,m)}$.

We evaluate each operation as a one-vs-rest classification problem, where the binary target is $g(i, c) = 1_{[y_i = c]}$ and the prediction score is $a(i, c)$. As a baseline, we repeat the same pipeline with randomly assigned labels and randomly selected token positions. We report AUROC and AUPRC report of middle token as representation position in the main text and other position in Appendix D.2, with more detailed implementation description.

Main finding: *Reasoning operations are representationally separable across all three models.*

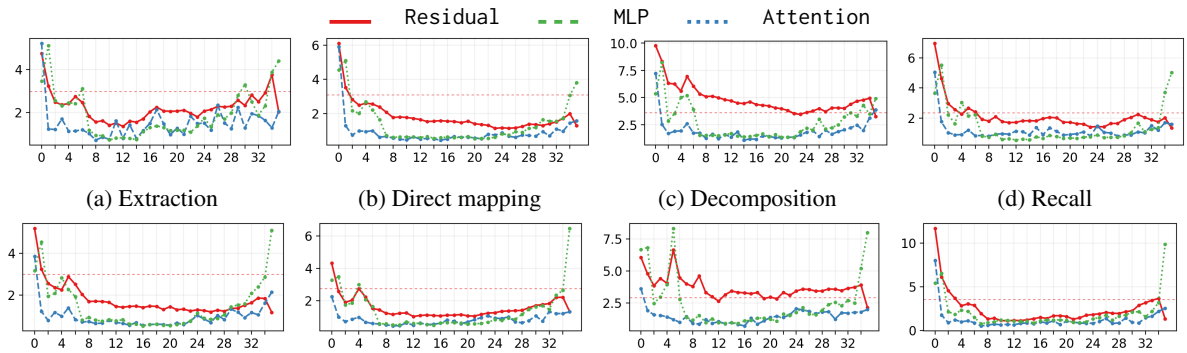
Figure 2 shows that operation-alignment scores are substantially higher than the random-label baseline for `Qwen2.5-7B`, `Qwen3-8B`, and `Gemma4-31B`,



(a) Extraction (b) Direct mapping (c) Decomposition (d) Recall

(e) Deduction (f) Algebraic Manipulation (g) Arithmetic Computation (h) Final Answer

Figure 3: **Visualization of reasoning-operation clusters in the learned LDA space.** For each target operation, held-out span representations from `Qwen3-8B` are projected onto a two-dimensional plane whose x-axis is the corresponding reasoning-vector direction d_c and whose y-axis is the first principal component of the residual representations. Points are colored by their annotated reasoning operation labels.



(a) Extraction (b) Direct mapping (c) Decomposition (d) Recall

(e) Deduction (f) Algebraic Manipulation (g) Arithmetic Computation (h) Final Answer

Figure 4: **Quantitative analysis on intra-span LDA score variances.** We investigate the LDA score variances within each spans across layers using Qwen3-8B. We further generalize the results for Qwen2.5-7B and Gemma4-31b in Appendix.

indicating that taxonomy-defined reasoning operations correspond to distinguishable structures in hidden space.

Separability also shows a consistent layer-wise pattern: it increases from early to middle layers and slightly decreases in later layers. This suggests that operation-level information is most strongly organized in intermediate representations, while later layers may become more specialized for next-token generation. We additionally verify with various token positions within reasoning spans. Middle- and last-token representations generally outperform first-token representations, while mean-pooled representations achieve the strongest separability.

Visualization. We further visualize the test spans in the learned LDA space. For each reasoning type, we define a 2D projection where the x-axis is the corresponding reasoning vector direction d_c, and the y-axis is the first principal component of the residual chunk representations after subtracting the projection onto d_c. All test spans are projected onto this plane and colored by their true operation labels. Figure 3 shows the resulting per-label projections for `Qwen3-8B`. Overall, the visualization shows label-dependent structure with moderate separability across reasoning operations, consistent with the AUROC and AUPRC results.

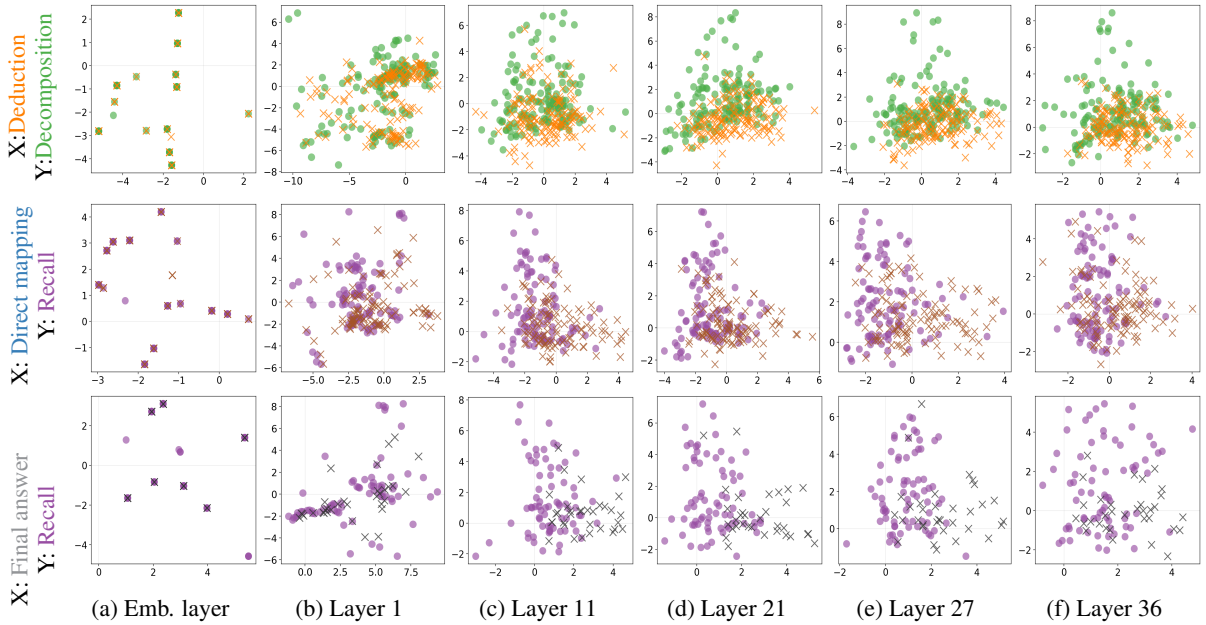


Figure 5: **Identical tokens acquire reasoning-dependent representations.** For pairs of reasoning operations, we collect common surface tokens that appear in both operation contexts and visualize their layer-wise scores along the two corresponding reasoning-vector directions in `Qwen3-8B`. Each point is an occurrence of a shared token and is colored by its annotated reasoning label.

3.3 How Reasoning-Operation Signals Evolve Across Layers

The previous section showed that reasoning operations are separable in hidden space, but separability varies substantially across layers. This raises several questions: why do some early layers already show nontrivial separability, are the signals only present in label specific words, and are later-layer signals merely inherited from easy-to-detect cue tokens in the autoregressive context?

3.3.1 Intra-span variance of operation scores

Experiment. To test whether early separability is driven by diagnostic tokens, we compute token-level operation scores within each annotated span. For a span labeled as operation c, we project each token representation onto the corresponding operation direction: $a_t^{(l,c)} = \left(z_t^{(l)}\right)^\top d_c^{(l)}$. We then measure how much this score varies across tokens in the span: $\mathrm{Var}_i^{(l)} = \mathrm{Var}_{t \in s_i}\left[a_t^{(l,y_i)}\right]$. High intra-span variance indicates that the operation signal is concentrated on a few tokens, whereas low variance indicates that the signal is distributed across the span.

Observation: *Early-layer separability is often sparse and token-local, while middle-layer separability is more distributed across the reasoning span.* We report the result in Figure 4. Empirically, intra-span variance is high in early layers, decreases toward middle layers, and increases slightly again in later layers. This suggests that shallow-layer separability can partly arise from sparse diagnostic tokens, while middle-layer separability reflects a more distributed and contextualized representation of the reasoning operation. Other model's per-operation variance trends are reported in Appendix A.

3.3.2 Identical Tokens Exhibit Different Score Trends Across Reasoning Contexts

Experiment. To examine whether reasoning vector scores reflect contextual reasoning states beyond lexical identity, we analyze identical tokens that appear across multiple reasoning contexts in the test set. We first collect balanced test spans for each reasoning label, using the same number of spans per label. Within each label, we count token frequencies over the selected span tokens. For each pair of reasoning labels, we then take the top-10 most frequent tokens from each label and select their intersection as the shared-token set. For every occurrence of these shared tokens in the corresponding test spans, we project the token representation at each layer into the LDA space and compute its score along the two label-specific probe directions.

Observation: *Common tokens that occur across multiple reasoning contexts still exhibit reasoning-*

Reasoning type	chunk #	$\Pr(\Delta < 0)$	Median Δ
Extraction	153	0.99	-12.14
Direct Mapping	107	0.95	-6.75
Decomposition	32	0.94	-10.77
Recall	205	0.94	-6.50
Deduction	104	0.80	-8.08
Algebraic manipulation	232	0.90	-11.10
Arithmetic computation	411	0.91	-8.67
Final answer	47	0.92	-3.00

Table 2: **Causal-position analysis of early cue peaks and late-layer signal onsets.** Negative Δ means that the late-layer operation signal appears before early-layer cue peaks.

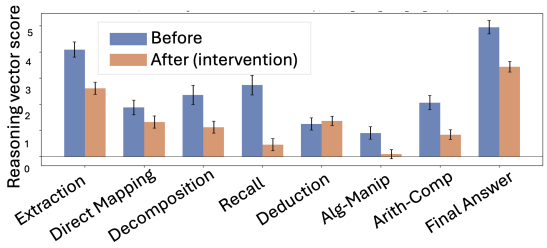


Figure 6: **Preceding context shapes operation representations.** Masking attention to preceding context reduces target operation scores, indicating that subsequent reasoning operations depend on prior reasoning context.

dependent score trends. Figure 5 shows layer-wise scatter plots for `Qwen3-8B`, where each point represents an occurrence of the same surface token and is colored by its true reasoning label. While the early-layer representations are largely intermixed, the points gradually shift in the middle-to-late layers, forming a mild separation trend along the two label-specific score directions. The final layer becomes more mixed again, suggesting that the clearest contextual reasoning signal emerges before the last layer. The shared tokens used to generate this plot are provided in Appendix F.

3.3.3 Are late-layer signals inherited from earlier cue tokens?

The variance analysis suggests that early-layer operation scores can be spiky, potentially reflecting local cue tokens. We therefore test whether later-layer separability simply reflects such earlier diagnostic tokens being carried forward through autoregressive attention. If this explanation were correct, early-layer cue peaks should precede the onset of the corresponding late-layer operation signal.

Experiment. For each reasoning trace and operation c, we identify early-layer peak positions $\mathcal{P}^c_{\text{early}}$ and the late-layer onset position t^c_{late}. The late onset is defined as the first position where a causal rolling mean of the late-layer score crosses a high-score threshold. We use a causal rolling mean because late-layer scores are less spiky and better characterized as sustained signals rather than isolated peaks. We then compute: $\Delta^c = t^c_{\text{late}} - \frac{1}{|\mathcal{P}^c_{\text{early}}|}\sum_{t\in\mathcal{P}^c_{\text{early}}} t$. If $\Delta^c > 0$, early cue peaks precede the late signal, which would be consistent with cue-token propagation. If $\Delta^c < 0$, the late signal appears before early cue peaks, making this explanation less plausible. We test whether the median of Δ^c is less than zero using a one-sided Wilcoxon signed-rank test.

Observation: *Late-layer operation signals are unlikely to be explained solely by earlier cue-token propagation.* We report the result in Table 2. Across models, reasoning types, and both full-trace and span-restricted variants, Δ is consistently negative and statistically significant. This suggests that late-layer separability is not merely inherited from earlier diagnostic tokens in the autoregressive context.

3.4 Preceding Contexts Trigger the Subsequent Reasoning Operations

Experiment. We intervene on held-out reasoning traces by re-running the model on the fixed generated sequence while masking attention from the first token of a target reasoning span to preceding context. We measure $\Delta_c = a_{\text{after}}(i, c) - a_{\text{before}}(i, c)$. We compare *pre-token* masking over the preceding M tokens, *pre-chunk* masking over the previous annotated span, and a random preceding-block control. We report pre-token results in Table 6; pre-chunk and random-control results are in Appendix C.

3.4.1 Result

Observation: *Preceding reasoning context shapes subsequent operation representations, with the strongest effect from the immediately preceding tokens.*

Across masking settings, blocking preceding context reduces the target operation score, indicating that operation-level representations are not formed independently within isolated chunks. The decrease is largest in the pre-token setting, followed by pre-chunk masking, while the random control shows the weakest change. This ordering suggests that the immediately preceding local context provides particularly strong cues for forming the next operation representation, whereas the broader pre-

vious span still contributes but with a more diffuse effect. The effect is not uniform across all labels: operations with smaller decreases may rely more on farther context or on information within the current span. Overall, these results indicate that subsequent reasoning operations are context-dependent, and that local preceding tokens play a particularly important triggering role.

4 Discussion

We further examine how reasoning-operation signals evolve over continuously generated tokens, rather than only at the level of annotated span representations. For selected reasoning traces, we compute token-level alignment scores with the corresponding reasoning-operation direction at every layer and visualize the resulting layer-by-token score maps. These heatmaps allow us to inspect whether an operation signal appears only at a few diagnostic tokens or is sustained across a broader generated segment. Qualitatively, we observe that operation-specific scores are often weak or localized in early layers, become more coherent across contiguous tokens in middle layers, and may become less sharply tied to the annotated operation in later layers. This trend is consistent with our span-level analyses: reasoning-operation representations are not merely attached to isolated lexical cues, but emerge through contextual processing over the generated reasoning trajectory. We provide representative heatmaps for Extraction, Recall, Decomposition, Deduction, and Arithmetic Computation in Figure C and D.

5 Related Work

Reasoning LLMs. Reasoning LLMs have progressed from eliciting explicit intermediate steps to structuring and training richer reasoning trajectories. Chain-of-thought prompting (Wei et al., 2022) improves arithmetic, commonsense, and symbolic reasoning by generating intermediate steps; ReAct (Yao et al., 2022) interleaves such reasoning traces with task-specific actions; and Tree-of-Thoughts (Yao et al., 2023) extends a single chain into search over multiple thought paths with self-evaluation, lookahead, and backtracking. More recent RL-based reasoning models such as DeepSeek-R1 (Guo et al., 2025) show that RL can elicit self-reflection, verification, and adaptive strategy use.

Mechanistic interpretability of reasoning LLMs. Several lines of work examine how reasoning is encoded in the internal representations of large language models. Latent and continuous reasoning methods demonstrate that reasoning need not be fully expressed through discrete natural-language tokens. Coconut (Hao et al., 2024) feeds the model's last hidden state back as a continuous thought, while CODI (Shen et al., 2025) and Soft-CoT (Xu et al., 2025b) compress or augment explicit chain-of-thought reasoning using continuous or soft representations. Representation-level analyses further reveal structured geometry: hidden-state probes predict answer correctness during reasoning (Zhang et al., 2025a), and trajectory-based studies show that reasoning steps occupy increasingly separable regions across layers, with correct and incorrect solutions diverging at later stages (Sun et al., 2026).

Reasoning operations and cognitive behaviors. Reasoning traces have also been analyzed through cognitive and behavioral decompositions. Classical problem-solving theory models reasoning in terms of problem states, operators, and search (Simon and Newell, 1971). Recent LLM studies operationalize this view by annotating or probing generated traces for distinct reasoning behaviors: CREST separates linear and non-linear behaviors such as verification and backtracking (Zhang et al., 2025b), while Kargupta et al. (2025) introduce a cognitive-science-grounded taxonomy of 28 reasoning elements and analyze their manifestations in LLM traces.

6 Conclusion

We investigated whether reasoning operations explicitly distinguished in text are organized as distinct geometric structures in the hidden representations of large language models. Through reasoning operation-based analysis, we found that reasoning operations form separable clusters in hidden space where even identical tokens acquire operation-specific representations depending on their surrounding reasoning context. Causal interventions further revealed that these operations are not locally self-contained but emerge through information propagation from preceding context. Consequently, our findings demonstrate that language models maintain a faithful representational correspondence between linguistic reasoning expressions and their internal geometric organization, offering a foundation for interpreting reasoning as a structured, layered, and context-dependent process.

7 Limitation

Our study has several limitations. First, the reasoning-operation labels are produced by LLM-based annotation. Although we use a detailed annotation prompt with explicit definitions and examples, these labels should be viewed as annotations of textually expressed reasoning functions rather than ground-truth labels of latent cognitive states. Future work should validate the taxonomy and span boundaries with human annotators and quantify inter-annotator agreement.

Second, our experiments are limited to mathematical and theorem-driven reasoning tasks and a small set of reasoning-oriented LLMs. While these settings provide structured reasoning traces, the observed geometry may differ in commonsense reasoning, planning, code generation, or interactive tasks, as well as in models with different architectures or training procedures.

Third, our analysis is primarily diagnostic rather than interventional or application-oriented. The learned reasoning-operation directions reveal representational separability and context dependence, but we do not evaluate whether they can be used to improve model behavior. For example, future work could investigate whether these probes can support reasoning failure detection, verification, decoding-time control, or activation-based steering.

8 Ethical Considerations

We used AI assistants during the preparation of this work. ChatGPT was used for writing support, including language polishing and drafting assistance, and Claude Code was used as a coding assistant during implementation. All experimental design, analyses, claims, and final manuscript content were reviewed and verified by the authors.

References

Wenhu Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony Xia. 2023. Theoremqa: A theorem-driven question answering dataset. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2024. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*.

Priyanka Kargupta, Shuyue Stella Li, Haocheng Wang, Jinu Lee, Shan Chen, Orevaoghene Ahia, Dean Light, Thomas L Griffiths, Max Kleiman-Weiner, Jiawei Han, and 1 others. 2025. Cognitive foundations for reasoning and their manifestation in llms. *arXiv preprint arXiv:2511.16660*.

George Polya. 1945. *How to solve it: A new aspect of mathematical method*. Princeton university press.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. 2025. CODI: Compressing chain-of-thought into continuous space via self-distillation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 677–693, Suzhou, China. Association for Computational Linguistics.

Herbert A Simon and Allen Newell. 1971. Human problem solving: The state of the theory in 1970. *American psychologist*, 26(2):145.

Lihao Sun, Hang Dong, Bo Qiao, Qingwei Lin, Dongmei Zhang, and Saravan Rajmohan. 2026. Llm reasoning as trajectories: Step-specific representation geometry and correctness signals. *arXiv preprint arXiv:2604.05655*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Fengli Xu, Qianyue Hao, Chenyang Shao, Zefang Zong, Yu Li, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, and 1 others. 2025a. Toward large reasoning models: A survey of reinforced reasoning with large language models. *Patterns*, 6(10).

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. 2025b. SoftCoT: Soft chain-of-thought for efficient reasoning with LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23336–23351, Vienna, Austria. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others.

2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, YuYue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, and 17 others. 2026. DAPO: An open-source LLM reinforcement learning system at scale. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. 2025a. Reasoning models know when they're right: Probing hidden states for self-verification. *arXiv preprint arXiv:2504.05419*.

Zhenyu Zhang, Xiaoxia Wu, Zhongzhu Zhou, Qingyang Wu, Yineng Zhang, Pragaash Ponnusamy, Harikaran Subbaraj, Jue Wang, Shuaiwen Leon Song, and Ben Athiwaratkun. 2025b. Understanding and steering the cognitive behaviors of reasoning models at test-time. *arXiv preprint arXiv:2512.24574*.

Appendix

- § A: Generalization of Key Findings across LLM Families
- § B: Token Reasoning Vector Score Heatmap
- § C: Additional Context-Intervention Results
- § D: Detailed Setup and Results for Representational Separability
- § E: Full Taxonomy of Reasoning Operations
- § F: Shared Tokens Used for Pairwise Analysis
- § H: Prompt Used in Annotation

A Generalization of Key Findings across LLM Families

We additionally examine whether the intra-span variance trend observed in `Qwen3-8B` also appears in other model families. Figures A and B show the per-operation variance trends for `Qwen2.5-7B` and `Gemma4-31B`, respectively. Overall, both models exhibit the same qualitative pattern: intra-span variance is relatively high in early layers, decreases toward middle layers, and slightly increases again in later layers. This indicates that the reduction of intra-span LDA-score variance is a general phenomenon across LLM families rather than a model-specific artifact.

Consistent with the main-text analysis, this trend suggests that early-layer separability is often supported by sparse and token-local cues, whereas middle-layer representations become more distributed across the reasoning span. The fact that the same pattern appears across multiple models supports the robustness of our interpretation of reasoning-operation signal formation.

B Token-level Reasoning Vector Score Heatmaps

In addition to span-level analyses, we visualize how reasoning-operation scores evolve over continuously generated tokens. For selected reasoning traces, we compute token-level alignment scores with the corresponding reasoning-operation direction at every layer and visualize the resulting layer-by-token score maps.

These heatmaps provide a qualitative view of whether an operation signal is concentrated on a small number of diagnostic tokens or sustained across a broader generated segment. Across the examples, operation-specific scores are often weak or localized in early layers, become more coherent across contiguous tokens in middle layers, and may become less sharply tied to the annotated operation in later layers. This pattern is consistent with our span-level separability and intra-span variance analyses, suggesting that reasoning-operation representations emerge through contextual processing rather than being attached only to isolated lexical cues.

The examples also show that identical surface tokens can receive different operation-score intensities depending on their surrounding reasoning context. For instance, in Figure D, the comma token appears multiple times in the same trace, but its score intensity differs across occurrences. This supports the interpretation that token-level reasoning-operation scores are modulated by contextualized reasoning states, rather than determined solely by token identity.

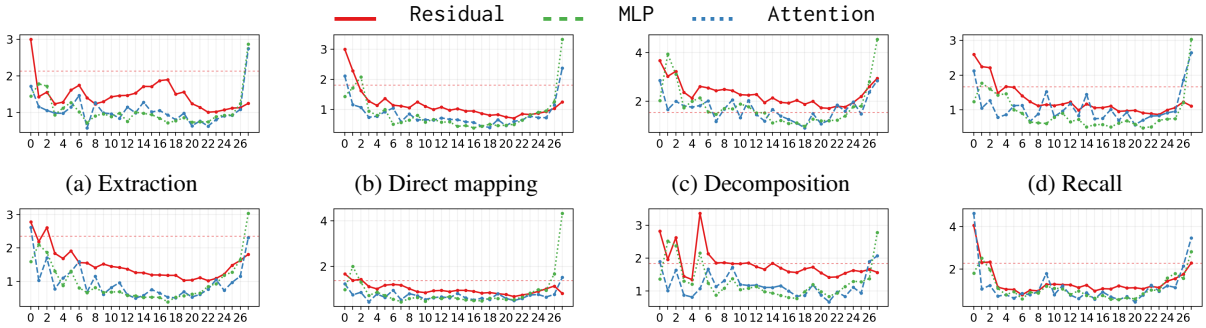
C Additional Context-Intervention Results

As shown in Figure E, the additional intervention results show the same ordering as the main pre-token result. The magnitude of the score decrease is generally smallest under the random preceding-block control, larger under pre-chunk masking, and largest under pre-token masking:

$$\text{random} < \text{pre-chunk} < \text{pre-token}.$$

This ordering suggests that the formation of a target reasoning-operation representation depends most strongly on the immediately preceding local context, while the previous reasoning span still contributes meaningful but more diffuse information. In contrast, randomly selected preceding blocks have a weaker effect, indicating that the score reduction is not simply caused by removing arbitrary past tokens.

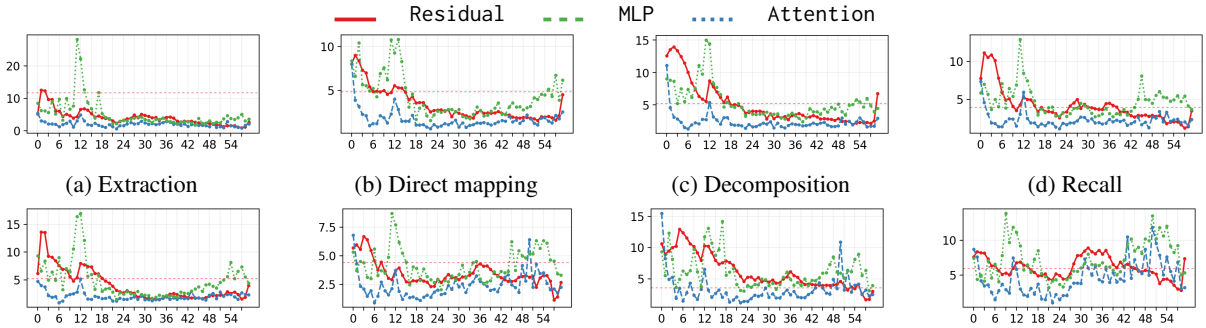
One exception is RECALL, where the random-control intervention produces a relatively large decrease. This may indicate that recall operations depend on information accumulated across a broader preceding context, rather than only on the immediately adjacent span or local transition tokens. Since recall often retrieves formulas, definitions, or theorems that can be cued by earlier problem statements or previously established subgoals, masking even a random preceding block can remove



(a) Extraction (b) Direct mapping (c) Decomposition (d) Recall

(e) Deduction (f) Algebraic Manipulation (g) Arithmetic Computation (h) Final Answer

Figure A: . intra-span LDA score variances of Qwen2.5-7B



(a) Extraction (b) Direct mapping (c) Decomposition (d) Recall

(e) Deduction (f) Algebraic Manipulation (g) Arithmetic Computation (h) Final Answer

Figure B: . intra-span LDA score variances of Gemma4-31B

context that remains relevant for forming the recall representation. Nevertheless, the overall trend across operation types supports the main conclusion: subsequent reasoning-operation representations are context-dependent, and structured preceding context plays a stronger role than arbitrary preceding tokens.

D Detailed Setup and Results for Representational Separability

D.1 Setup Description

This section provides additional implementation details for the representational separability analysis in Section 3.2.

Representation extraction. For each model and dataset, we generate reasoning traces and retain only responses that lead to correct final answers. This allows us to analyze operation-level representation geometry under successful reasoning trajectories. For every retained response, we save the generated token sequence and extract hidden representations from each layer. For token position t, layer l, and representation type m, we denote the hidden representation as

$$h_t^{(l,m)} \in \mathbb{R}^d,$$

where

$$m \in \{\mathrm{attn}, \mathrm{mlp}, \mathrm{res}\}$$

indicates the attention output, MLP output, and residual stream output after the layer, respectively. We additionally include the embedding layer as $l = 0$.

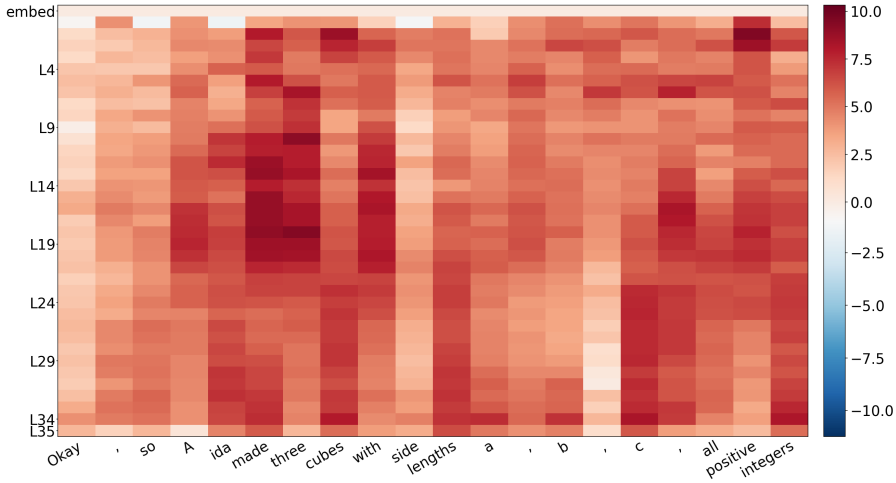
Reasoning-span annotation. Each generated reasoning trace is segmented into non-overlapping reasoning spans. A span is represented as

$$s_i = \left(t_i^{\mathrm{start}}, t_i^{\mathrm{end}}, y_i\right),$$

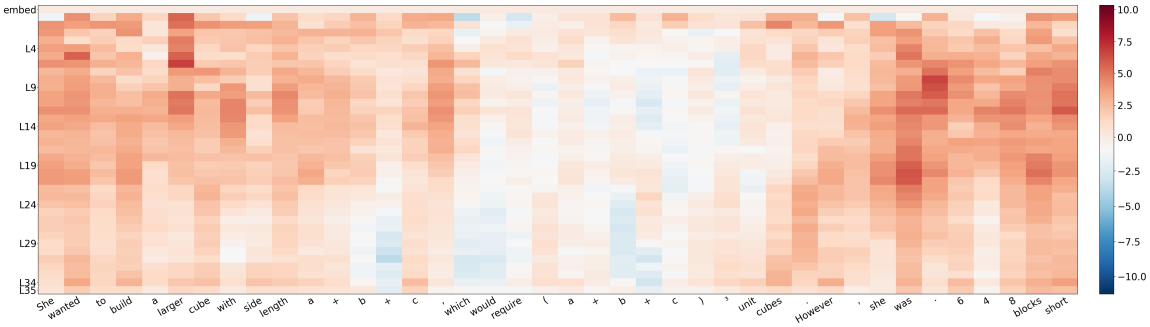
where t_i^{start} and t_i^{end} denote the inclusive token boundaries of the span, and

$$y_i \in \mathcal{Y}$$

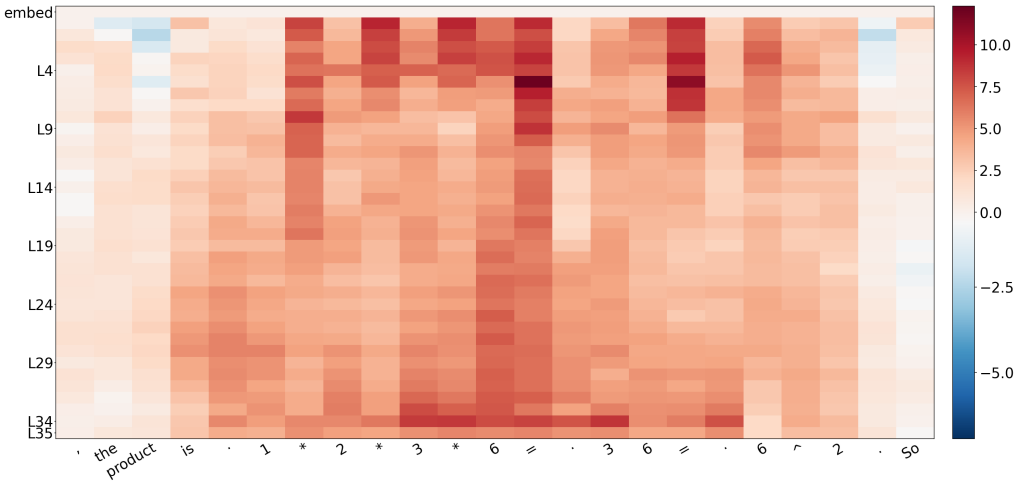
is the assigned reasoning-operation label. In the main analysis, $\mathcal{Y}$ contains the eight recurring operation labels: Extraction, Direct mapping, Decomposition, Recall, Deduction, Algebraic manipulation, Arithmetic computation, and Final answer. Since



(a) Extraction vector - Example 1

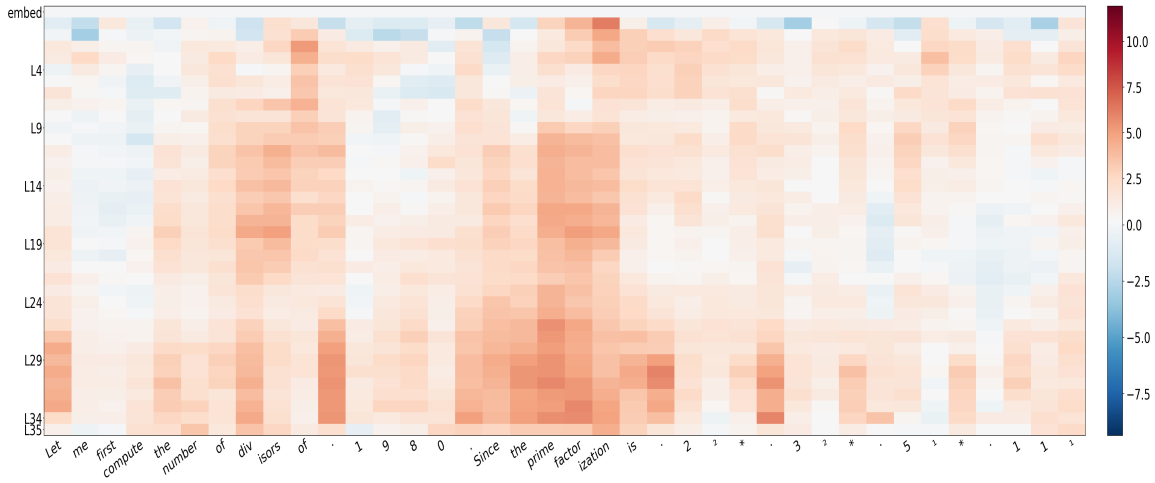


(b) Extraction vector - Example 2

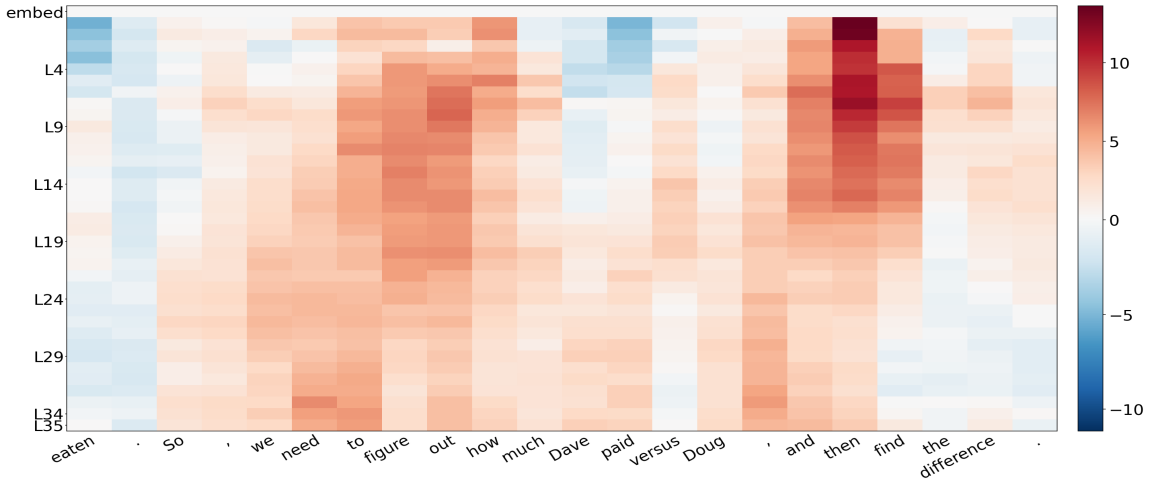


(c) Arithmetic Computation

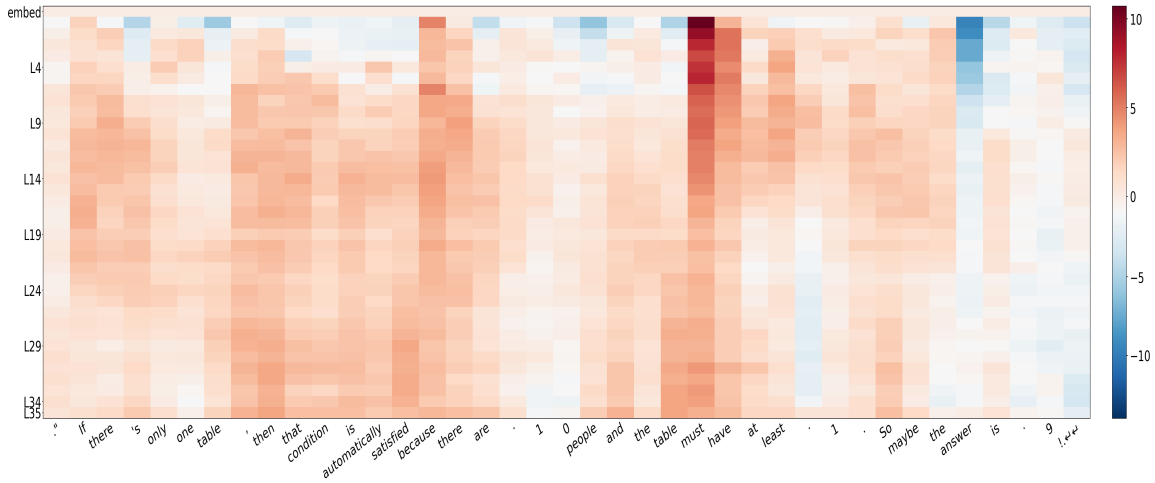
Figure C: **Layer-by-token heatmaps of reasoning-operation alignment scores.** We visualize token-level alignment scores with the corresponding reasoning-operation direction across layers for representative reasoning traces. Each heatmap shows how strongly each generated token aligns with a target operation direction at each layer. The examples illustrate that operation-specific scores are often localized or weak in early layers, become more coherent over contiguous token spans in middle layers, and may become less sharply localized in later layers. This supports the view that reasoning-operation representations emerge through contextual processing rather than being attached only to isolated diagnostic tokens.



(a) Recall

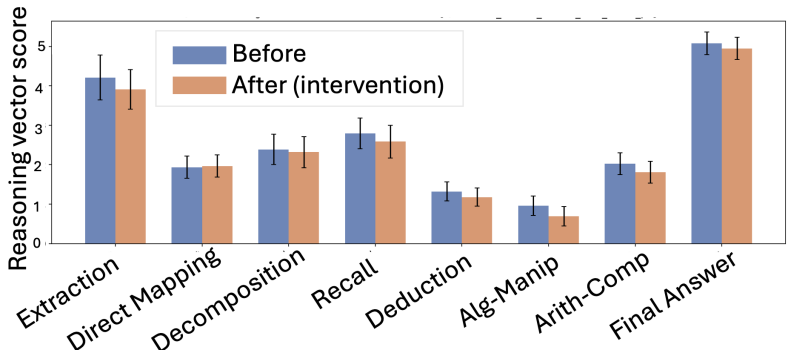


(b) Decomposition

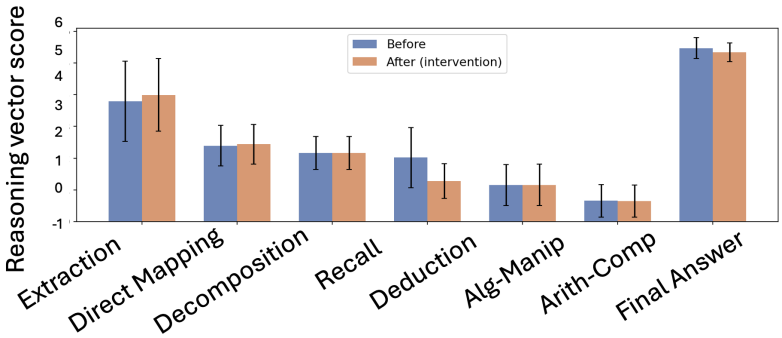


(c) Deduction

Figure D: **Additional layer-by-token heatmaps of reasoning-operation alignment scores.** We provide additional representative examples for other reasoning-operation types. Consistent with Figure C, the alignment scores become more structured across contiguous generated tokens in middle layers, while early-layer scores are more sparse and token-local. These qualitative patterns are consistent with the span-level separability and intra-span variance analyses.



(a) Pre-chunk masking



(b) Random preceding-block control

Figure E: **Additional attention-masking intervention results.** We report context-intervention results beyond the pre-token setting shown in the main text. In the pre-chunk setting, we mask attention from the first token of a target reasoning span to the entire immediately preceding annotated reasoning span. In the random-control setting, we mask a randomly selected preceding block with a comparable length. Across settings, blocking preceding context generally reduces the target operation score, indicating that subsequent reasoning-operation representations depend on prior context. The reduction is stronger for structured preceding-context masking than for the random control, supporting the view that relevant preceding reasoning context contributes to the formation of the next operation representation.

these annotations are obtained from generated text, they should be interpreted as labels of textually expressed reasoning functions rather than direct labels of latent cognitive states. The number of spans generated are in Table D.

Span-level representation. Because a reasoning span may contain multiple tokens, we construct a span-level representation from the token representations inside the span. We consider four pooling strategies: first-token, middle-token, last-token, and mean pooling. For a span s_i, layer l, and representation type m, the span representation is

$$x_i^{(l,m)} = \mathrm{Pool}\left(\left\{h_t^{(l,m)} : t \in [t_i^{\mathrm{start}}, t_i^{\mathrm{end}}]\right\}\right),$$

where $\mathrm{Pool}(\cdot)$ is one of the four pooling strategies above. The main results use the middle-token representation, while the other pooling variants are reported separately.

Entropy-centered span window. For the span-level analysis, we use a fixed-window representation to avoid very long spans dominating the pooled representation. For each annotated span shorter than 300 tokens, we first identify the token position with the highest generation entropy:

$$u_i = \arg\max_{t \in [t_i^{\mathrm{start}}, t_i^{\mathrm{end}}]} H_t,$$

where H_t is the token-level generation entropy at position t. We then select a 50-token window centered at u_i. If the window exceeds the span boundary, we shift it so that the full window remains inside the annotated span. Spans longer than 300 tokens are excluded from this analysis. This windowing strategy is intended to capture the region where the model is relatively uncertain or computationally active within the reasoning span.

Let the resulting token window be denoted by

$$W_i \subseteq [t_i^{\mathrm{start}}, t_i^{\mathrm{end}}].$$

The pooled span representation is then computed as

$$x_i^{(l,m)} = \mathrm{Pool}\left(\left\{h_t^{(l,m)} : t \in W_i\right\}\right).$$

Projection into a discriminative subspace. For each layer l and representation type m, we split annotated spans into train and test sets. All normalization and projection parameters are fitted only on the train split and then applied to the held-out test split.

We first apply L_2 normalization:

$$\tilde{x}_i^{(l,m)} = \frac{x_i^{(l,m)}}{\left\|x_i^{(l,m)}\right\|_2}.$$

We then apply PCA to reduce the representation dimension to $K = 128$, followed by supervised Linear Discriminant Analysis (LDA). Since the number of operation classes is $C = 8$, the LDA subspace has at most $C - 1 = 7$ dimensions. The projected representation is

$$z_i^{(l,m)} = W_{\mathrm{LDA}}^{\top} W_{\mathrm{PCA}}^{\top} \tilde{x}_i^{(l,m)}.$$

Both W_{PCA} and W_{LDA} are fitted using only the train split.

Operation prototypes. For each reasoning operation $c \in \mathcal{Y}$, we compute the centroid of projected train representations belonging to class c:

$$\mu_c^{(l,m)} = \frac{1}{|\mathcal{D}_c|} \sum_{i \in \mathcal{D}_c} z_i^{(l,m)},$$

where

$$\mathcal{D}_c = \{i : y_i = c\}$$

denotes the set of train spans labeled as operation c.

We also compute the centroid of all other train spans:

$$\mu_{\neg c}^{(l,m)} = \frac{1}{|\mathcal{D}_{\neg c}|} \sum_{i \in \mathcal{D}_{\neg c}} z_i^{(l,m)},$$

where

$$\mathcal{D}_{\neg c} = \{i : y_i \neq c\}.$$

One-vs-rest operation direction. We define the one-vs-rest operation direction for class c as the normalized difference between the class centroid and the rest centroid:

$$d_c^{(l,m)} = \frac{\mu_c^{(l,m)} - \mu_{\neg c}^{(l,m)}}{\left\|\mu_c^{(l,m)} - \mu_{\neg c}^{(l,m)}\right\|_2}.$$

This direction captures the axis in the LDA-projected space that separates operation c from the remaining reasoning operations.

Alignment score. For each held-out test span i and reasoning operation c, we compute the dot-product alignment between the projected span representation and the operation direction:

$$a(i,c) = \left(z_i^{(l,m)}\right)^{\top} d_c^{(l,m)}.$$

A larger value of $a(i,c)$ indicates that the span representation is more strongly aligned with the direction associated with operation c.

One-vs-rest evaluation. For each operation c, we formulate a one-vs-rest binary classification problem. The binary target is

$$g(i,c) = \mathbb{1}[y_i = c],$$

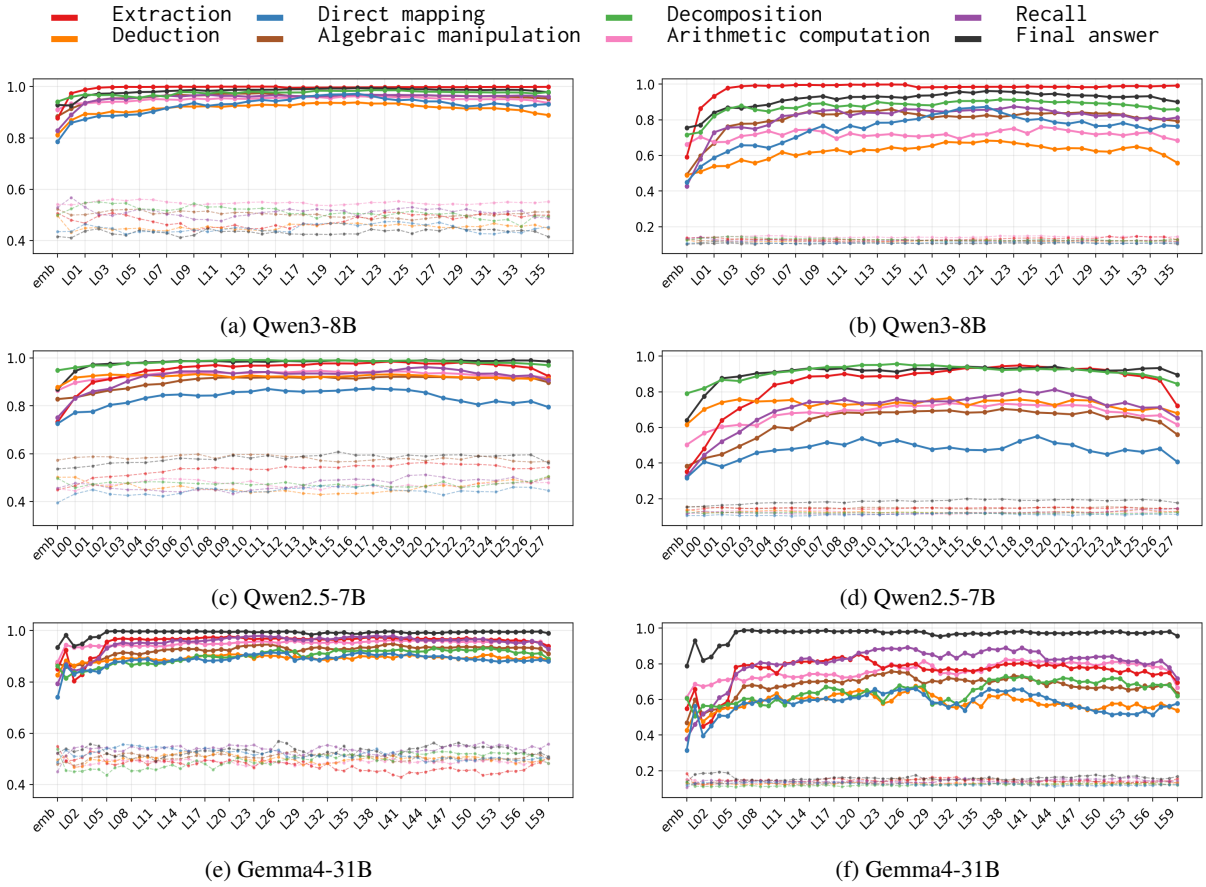
and the prediction score is the alignment score $a(i,c)$. We compute AUROC and AUPRC by varying the decision threshold over $a(i,c)$. A high AUROC indicates that spans labeled as operation c tend to have higher alignment with the operation direction $d_c^{(l,m)}$ than spans labeled as other operations.

Random baseline. As a control, we repeat the same evaluation pipeline with randomly assigned operation labels and randomly selected token positions. The random baseline uses the same number of spans and the same train-test protocol as the main experiment. This baseline tests whether the observed separability arises from the reasoning-operation annotation rather than from the projection pipeline or dataset imbalance alone.

D.2 Detailed Results for Span Position Variants

In the main text, we report separability results using the middle-token representation of each reasoning span. Here, we provide additional results using mean-pooled, first-token, and last-token span representations. Overall, the results are qualitatively consistent with the middle-token setting: reasoning-operation labels remain substantially more separable than the random-label baseline across models and layers, indicating that the observed operation-level geometry is not specific to a single choice of span position.

The first-token and last-token settings show small quantitative differences, but they largely preserve the same layer-wise trend observed in the



(a) Qwen3-8B (b) Qwen3-8B

(c) Qwen2.5-7B (d) Qwen2.5-7B

(e) Gemma4-31B (f) Gemma4-31B

Figure F: **Quantitative analyses on separability of reasoning operations in representation spaces at position mean.**

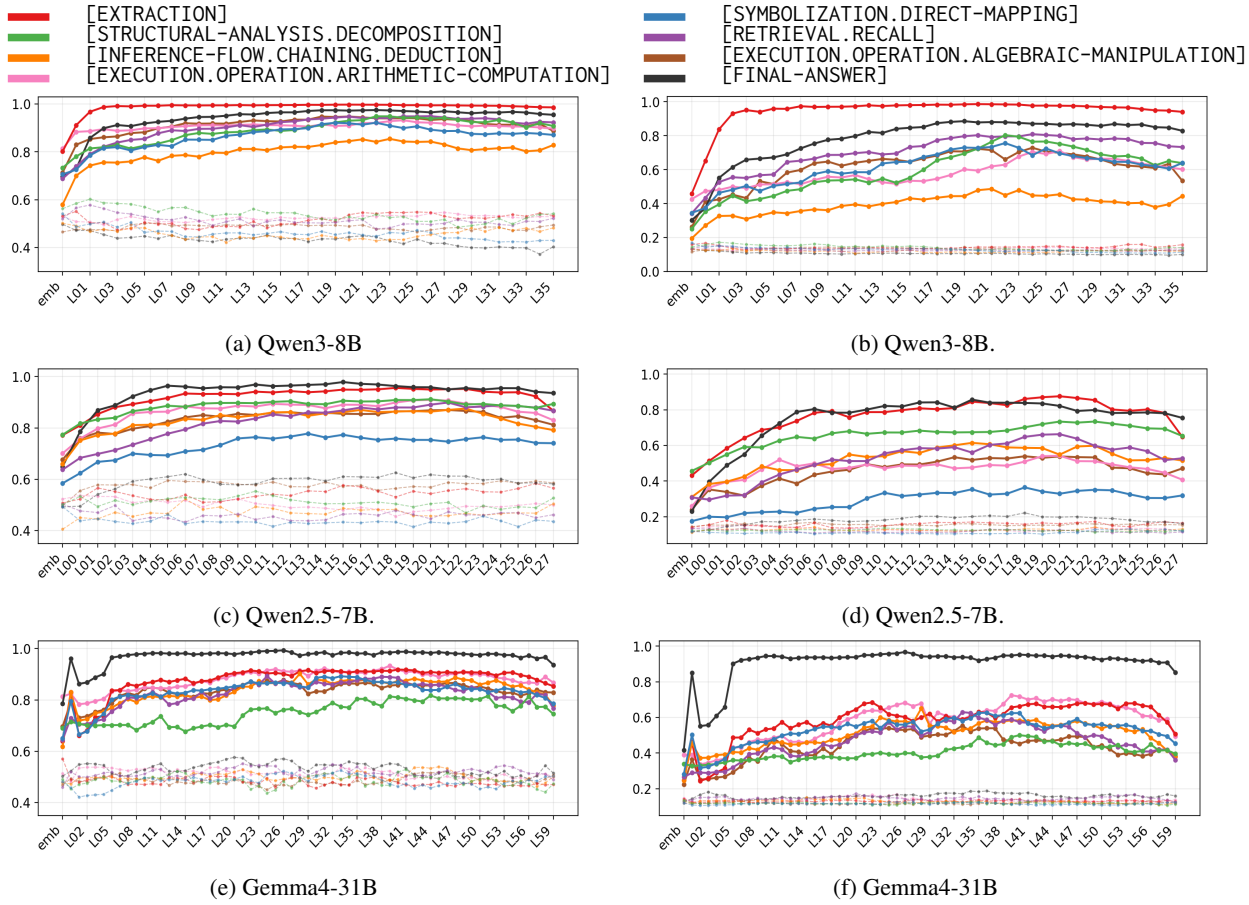
main results. Separability generally increases from early layers to middle layers and remains above the random baseline across most operation types. This suggests that operation-level information is available at multiple positions within a reasoning span, although single-token representations can be more sensitive to local lexical content or boundary effects.

The mean-pooled representation shows a somewhat different pattern. Compared with single-token representations, mean pooling often yields stronger separability from earlier layers. This is expected because mean pooling aggregates information across multiple tokens in the span, which can make operation-level signals more stable and reduce the effect of isolated token-level noise. As discussed in Section 3.3, early-layer operation scores can be sparse and token-local, whereas middle-layer scores become more distributed across the span. The stronger early-layer performance of mean pooling is therefore consistent with the intra-span variance analysis: aggregating over tokens can capture multiple local cues even before the operation signal becomes fully distributed in middle layers.

Taken together, these position-variant results support the robustness of our main finding. Reasoning operations are geometrically separable not only at the middle token, but also under alternative choices of span representation. At the same time, the differences between single-token and mean-pooled representations indicate that the measured separability depends partly on how much within-span context is aggregated.

E Full Taxonomy of Reasoning Operations

The main analysis focuses on eight recurring reasoning-operation types that appear frequently enough for representation-level analysis. At Table A–C, we provide the full hierarchical taxonomy used for annotation, including additional operation types and subtypes that were used to organize the annotation schema but were not included in the main separability experiments. The taxonomy follows the four problem-solving stages introduced



(a) Qwen3-8B (b) Qwen3-8B.

(c) Qwen2.5-7B. (d) Qwen2.5-7B.

(e) Gemma4-31B (f) Gemma4-31B

Figure G: **Quantitative analyses on separability of reasoning operations in representation spaces at position first.**

in the main text and specifies each operation by its functional role in the generated reasoning trace.

F Shared Tokens Used for Pairwise Analysis

In Section 3.3, we analyze whether identical surface tokens acquire different reasoning-operation scores depending on their surrounding reasoning context. For each pair of reasoning-operation labels, we first collect the most frequent tokens from balanced test spans of each label and then take their intersection. The resulting shared-token set is used to compare token representations while controlling for lexical identity.

In the main text, we visualize three representative operation pairs: DEDUCTION vs. DECOMPOSITION, DIRECT MAPPING vs. RECALL, and FINAL ANSWER vs. RECALL. The shared tokens used for these pairs are listed below:

DEDUCTION vs. DECOMPOSITION : {a, is, of, so, that, the, to},

DIRECT MAPPING vs. RECALL : {a, is, let, of, so, that, the, to},

FINAL ANSWER vs. RECALL : {is, of, so, that, the, to}.

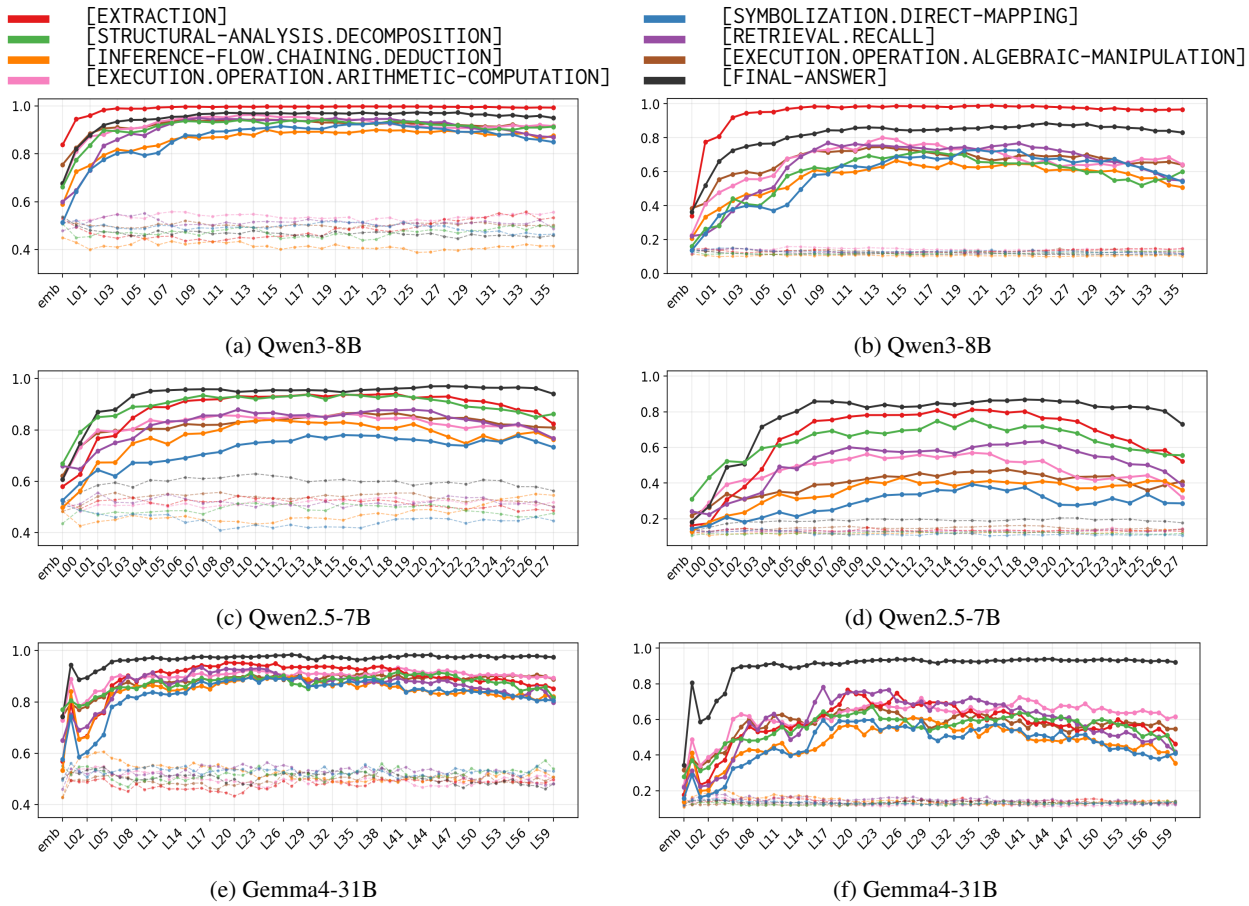
These tokens are mostly function words or common reasoning connectives. Therefore, any separation observed in the LDA score space is unlikely to be explained by token identity alone.

G Computation environment

For each model and dataset, we generate reasoning traces and collect hidden representations corresponding to the labeled reasoning steps. Inference is performed using eight NVIDIA V100 GPUs, and subsequent representation analyses are conducted on a single V100 node.

H Prompt used in Annotation

We provide the full prompt used to annotate generated reasoning traces into reasoning-operation spans. The prompt provides the full taxonomy, span-selection rules, output JSON format, and examples for standardizing the annotation procedure.



(a) Qwen3-8B (b) Qwen3-8B

(c) Qwen2.5-7B (d) Qwen2.5-7B

(e) Gemma4-31B (f) Gemma4-31B

Figure H: **Quantitative analyses on separability of reasoning operations in representation spaces at position last.**

Table A: Reasoning type schema (Stages 1–2).

Reasoning stage	Level 1	Level 2	Level 3	Reasoning type	Description / Example
Stage 1 Understanding the Problem	Extraction	–	–	Extraction	Directly retrieving given information from the problem statement without modification. *Example:* "Two numbers add up to 10, and both are non-negative."
	Symbolization	Direct-mapping	–	Direct-mapping	Directly converting a statement into a formal representation without changing its structure. *Example:* The sum of two numbers is 10 $\rightarrow x + y = 10$.
		Reframing	–	Reframing	Reconstructing the problem in a different representational system. *Example:* Geometry problem $\rightarrow$ coordinate geometry.
		Abstraction	–	Abstraction	Removing unnecessary details and retaining only the essential structure. *Example:* 3 red balls and 2 blue balls $\rightarrow (3, 2)$.
	State-space-definition	–	–	State-space-definition	Defining possible states, constraints, and the solution space. *Example:* $x \geq 0, y \geq 0, x + y = 10$.
Stage 2 Planning the Solution	Pattern-matching	–	–	Pattern-matching	Mapping the current problem to a known problem type, template, or strategy. *Example:* Flip a coin three times $\rightarrow$ probability/combinatorics problem.
	Structural-analysis	Symmetry	–	Symmetry	Identifying whether the problem remains unchanged under transformations. *Example:* Normal distribution is symmetric about the mean $\rightarrow$ compute one side and double it.
		Invariance	–	Invariance	Identifying quantities that remain unchanged throughout a process. *Example:* Energy conservation $\rightarrow E_{\text{initial}} = E_{\text{final}}$.
		Decomposition	–	Decomposition	Breaking a complex problem into simpler or independent subproblems. *Example:* Heptagon $\rightarrow$ divide into triangles $\rightarrow$ solve each triangle.
	Idealization	–	–	Idealization	Simplifying the problem by removing irrelevant or noisy details. *Example:* Motion with friction $\rightarrow$ ignore friction.
	Hypothesis-formation	–	–	Hypothesis-formation	Generating a conjecture or assumption based on observed patterns or partial reasoning. *Example:* Terms cancel out $\rightarrow$ the result may be 0.

Table B: Reasoning type schema (Stage 3).

Reasoning stage	Level 1	Level 2	Level 3	Reasoning type	Description / Example
Stage 3 Carrying Out the Plan	Retrieval	Recall	–	Recall	Retrieving relevant definitions, rules, concepts, or formulas. *Example:* Need area of a circle $\rightarrow A = \pi r^2$.
		Matching	–	Matching	Determining whether retrieved knowledge applies to the current situation. *Example:* Apply $A = \pi r^2$ when the radius is given.
	Inference-flow	Chaining	Deduction	Deduction	Deriving a specific conclusion from a general rule. *Example:* General theorem $\rightarrow$ apply to the given case.
			Induction	Induction	Inferring a general rule from observed patterns. *Example:* Several terms follow the same difference $\rightarrow$ infer arithmetic pattern.
			Analogy	Analogy	Reasoning from a structurally similar case. *Example:* Solve a new problem using a similar solved template.
		Branching	–	Branching	Splitting into cases and analyzing each separately. *Example:* $x^2 = 4 \rightarrow x = 2$ or $x = -2$.
	Execution	Instantiation	–	Instantiation	Assigning specific values or conditions to abstract variables or expressions. *Example:* $x + y = 10$, let $x = 3$, then $y = 7$.
		Operation	Algebraic-manipulation	Algebraic-manipulation	Transforming expressions or equations. *Example:* $x + y = 10 \rightarrow y = 10 - x$.
			Arithmetic-computation	Arithmetic-computation	Performing numerical calculations. *Example:* $10 - 3 = 7$.
			Logical-evaluation	Logical-evaluation	Evaluating truth values or logical implications. *Example:* If $P \rightarrow Q$ and P is true, then Q is true.
	Representation-construction	–	–	Representation-construction	Organizing computed results into structured forms such as lists, sequences, or tables. *Example:* Possible values: $(1, 9), (2, 8), (3, 7), \ldots$
	Pattern-extraction	–	–	Pattern-extraction	Identifying patterns, regularities, or trends from computed results. *Example:* $2, 4, 6, 8 \rightarrow$ increase by 2.

Table C: Reasoning type schema (Stage 4).

Reasoning stage	Level 1	Level 2	Level 3	Reasoning type	Description / Example
Stage 4 Looking Back and Final Answer	Strategy-validation	–	–	Strategy-validation	Verifying whether the chosen strategy or assumptions were valid. *Example:* Check whether differences are constant before assuming an arithmetic sequence.
	Error-detection	–	–	Error-detection	Identifying mistakes in reasoning, calculation, or logical steps. *Example:* $10 - 3 = 6 \rightarrow$ incorrect arithmetic; should be 7.
	Dimensional-analysis	–	–	Dimensional-analysis	Checking whether units or dimensions are consistent with the required quantity. *Example:* Speed = distance/time $\rightarrow$ meters/second.
	Extreme-case-testing	–	–	Extreme-case-testing	Testing whether the solution holds under extreme or boundary conditions. *Example:* For $f(x) = x/(x+1)$, test $x = 0$ and $x \rightarrow \infty$.
	Final-answer	–	–	Final-answer	Clearly presenting the final result in the required format. *Example:* Therefore, $x = 3$ and $y = 7$.

Table D: Train + Test occurrence counts by reasoning type and model.

Stage	Reasoning type	Qwen2.5-7B	Qwen3-8B	Gemma4-31B
Stage 1: Understanding the Problem	Extraction	502	464	217
	Direct-mapping	556	396	429
	Reframing	130	188	160
	Abstraction	8	18	13
	State-space-definition	162	149	83
Stage 2: Planning the Solution	Pattern-matching	186	144	113
	Symmetry	19	58	33
	Invariance	14	18	22
	Decomposition	374	135	96
	Idealization	11	9	5
	Hypothesis-formation	46	64	26
Stage 3: Carrying Out the Plan	Recall	704	844	515
	Matching	91	69	43
	Deduction	357	576	427
	Induction	2	2	4
	Analogy	2	–	–
	Branching	69	70	88
	Instantiation	326	122	218
	Algebraic-manipulation	743	937	925
	Arithmetic-computation	626	1539	1316
	Logical-evaluation	215	299	261
	Representation-construction	74	158	138
	Pattern-extraction	5	19	15
Stage 4: Looking Back and Final Answer	Strategy-validation	53	246	40
	Error-detection	81	77	29
	Dimensional-analysis	2	19	–
	Extreme-case-testing	1	21	3
	Final-answer	354	306	271

System Prompt

```
You are given a numbered solution to a math/science problem.

Each word has been prefixed with §N (a zero-based global index):
  §0 word0 §1 word1 §2 word2 ...

Your task: identify reasoning spans by their §N word-index range and assign each a tag.

----------------------------------

Full description of tags:

# Stage 1: Understanding the Problem
## What this stage does
The Understanding stage is the process of transforming a problem from its original, often
↪  informal description into a structured and analyzable representation.
At this stage, the goal is not to solve the problem, but to:
- identify the relevant information,
- convert it into formal representations, and
- define the space in which the solution exists.

## Components
1. EXTRACTION
  [Definition] The process of directly retrieving the given information from the problem
  ↪  statement without modification.
  [Role]
  - Ensures that all relevant data is explicitly identified
  - Prevents omission or misinterpretation of key conditions
  [Example]
  - Two numbers add up to 10, and both are non-negative.
2. SYMBOLIZATION
  [Definition] The process of converting informal information (natural language, descriptions)
  ↪  into formal representations such as variables, equations, or logical structures.
  [Example]
  - The sum of two numbers is 10 -> x + y = 10
  [Subcomponents]
  2-1. SYMBOLIZATION.DIRECT-MAPPING
          [Definition] Directly converting the given statement into a formal representation
          ↪  without changing its structure or interpretation.
          [Example]
          - The sum of two numbers is 10 → x + y = 10
          - x is greater than 3 → x > 3
          - A number is even → x = 2k
  2-2. SYMBOLIZATION.REFRAMING
    [Definition] Reconstructing the problem in a different representational system (e.g.,
    ↪  algebraic, geometric, probabilistic).
    [Example]
    - Geometry problem → coordinate geometry
    - Word problem → algebraic equation
  2-3. SYMBOLIZATION.ABSTRACTION
    [Definition] Removing unnecessary details and retaining only the essential structure of the
    ↪  problem.
    [Example]
    - 3 red balls and 2 blue balls -> (3, 2) (Color becomes irrelevant; only quantities matter)
3. STATE-SPACE-DEFINITION
  [Definition] The process of defining the set of all possible states (solutions or
  ↪  configurations), and the constraints that restrict them
  [Core elements]
  - State variables (e.g., x, y)
  - Constraints (e.g., x  0, y  0)
  - Solution space (set of all valid states)
  [Role]
  - Determines where valid solutions can exist
  - Restricts the problem to a well-defined domain
  - Enables systematic reasoning and search
  [Example]
  - x  0, y  0
  - x + y = 10
```

```
# Stage 2: Planning the Solution
## What this stage does
The Planning stage is the process of determining how to solve the problem after it has been
↪  properly understood.
At this stage, the goal is to:
- identify relevant problem-solving strategies,
- analyze structural relationships within the problem, and
- propose a direction for execution.
Unlike the Understanding stage (which builds representation), Planning focuses on strategy
↪  selection and structural insight.
## Components
1. PATTERN-MATCHING
  [Definition] The process of mapping the current problem to a known problem type, template, or
  ↪  strategy.
  [Role]
  - Activates prior knowledge
  - Reduces problem complexity
  - Suggests a solution approach
  [Example]
  - Flip a coin three times → **probability + combinatorics problem**
  - Find the shortest path between points → **graph / optimization problem**
2. STRUCTURAL-ANALYSIS
  [Definition] The process of analyzing the internal structure, relationships, and invariant
  ↪  properties of the problem.
  [Core Idea] Focus on how elements within the problem are related and how those relationships
  ↪  constrain the solution.
  [Subcomponents]
  2-1. STRUCTURAL-ANALYSIS.SYMMETRY
        [Definition] Identifying whether the problem remains unchanged under certain
        ↪  transformations (e.g., reflection, rotation, permutation)
        [Example]
        Normal distribution
        → symmetric about the mean
        → **compute one side and double it**
        2-2. STRUCTURAL-ANALYSIS.INVARIANCE
        [Definition] Identifying quantities that remain unchanged throughout a process.
        [Example]
        Energy conservation
        → total energy remains constant
        → **E_initial = E_final**
        2-3. STRUCTURAL-ANALYSIS.DECOMPOSITION
        [Definition] Breaking a complex problem into simpler or independent subproblems.
        [Example]
        Heptagon → **divide into triangles → solve each triangle** → combine results
        2-4. IDEALIZATION
        [Definition] Simplifying the problem by removing irrelevant or noisy details.
        [Example]
        Motion with friction → **ignore friction** → idealized motion
3. HYPOTHESIS-FORMATION
        [Definition] The process of generating a conjecture or assumption based on observed
        ↪  patterns or partial reasoning.
        [Role]
        - Guides further reasoning
        - Narrows down possibilities
        - Suggests direction for execution
        [Example]
        Observed pattern: Terms cancel out → Hypothesis: The result may be 0
```

```
# Stage 3: Carrying Out the Plan
## What this stage does
The Carry Out stage is the process of executing the planned strategy to produce concrete results.
At this stage, the goal is to:
- apply relevant knowledge and rules,
- perform logical and numerical operations,
- explore possible cases, and
- construct intermediate and final results.
Unlike the Planning stage (which determines what to do), this stage focuses on actually
↪  performing the reasoning and computations.
## Components
1. RETRIEVAL
  [Definition] The process of recalling relevant concepts, definitions, rules, or formulas from
  ↪  prior knowledge.
  [Role]
  - Provides the necessary tools for execution
  - Connects abstract knowledge to the current problem
  [Subcomponents]
  2-1. RETRIEVAL.RECALL: retrieving definitions or formulas
  2-2. RETRIEVAL.MATCHING: determining whether the knowledge applies to the current situation
  [Structure]
        - [Problem context] → [Relevant concept retrieved] → [Apply if applicable]
        [Example]
        - Area of a circle is needed -> **A = r²** -> apply to problem
2. INFERENCE-FLOW
  [Definition] The process of constructing a sequence of logical steps (chaining) or dividing the
  ↪  problem into multiple cases (branching).
  [Role]
  - Enables step-by-step reasoning
  - Handles conditional or multiple-case scenarios
  [Subcomponents]
  2-1. CHAINING
    2-1-1. CHAINING.DEDUCTION (general → specific)
    2-1-2. CHAINING.INDUCTION (pattern → general rule)
    2-1-3. CHAINING.ANALOGY (similar structure)
  2-2. BRANCHING: Splitting into cases and analyzing each separately
  [Structure]
  - Chaining: [A] → [B] → [C]
  - Branching: [Condition] → Case 1 / → Case 2 / → Case 3
  [Example]
  - x² = 4 → Branching: **x = 2** or **x = -2**
3. EXECUTION
  [Subcomponents]
  3-1. EXECUTION.INSTANTIATION
  [Definition] Assigning specific values or conditions to abstract variables or expressions.
  [Role]
  - Moves from abstract representation to concrete evaluation
  [Example]
  - x + y = 10 → Let x = 3 → Then y = 7
  3-2. EXECUTION.OPERATION: Performing actual computations or transformations on expressions.
    3-2-1. EXECUTION.OPERATION.ALGEBRAIC-MANIPULATION: Transforming the structure of expressions
    ↪  or equations.
   [Example]
   - x + y = 10 → y = 10 - x
    3-2-2. EXECUTION.OPERATION.ARITHMETIC-COMPUTATION: Performing numerical calculations.
   [Example]
   - 10 - 3 = 7
    3-2-3. EXECUTION.OPERATION.LOGICAL-EVALUATION: Evaluating truth values or logical
    ↪  implications.
    [Example]
    - If P → Q, and P is true → Q is true
4. REPRESENTAION-CONSTRUCTION
  [Definition] Organizing computed results into structured forms such as lists, sequences, or
  ↪  tables.
  [Role]
        - Makes results interpretable
        - Prepares for further reasoning or comparison
        [Example]
        - Possible values: (1,9), (2,8), (3,7), ...
5. PATTERN-EXTRACTION
  [Definition] Identifying patterns, regularities, or trends from computed results.
  [Role]
  - Enables generalization
  - Supports further reasoning or simplification
  [Example]
  - Sequence: 2, 4, 6, 8 → Pattern: **Increase by 2**
```

```
# Stage 4: Looking Back and Final Answer
## What this stage does
The Look Back stage is the process of evaluating and validating the solution after it has been
↪  obtained.
At this stage, the goal is to:
- verify correctness,
- detect possible errors,
- test robustness under different conditions, and
- improve or refine the solution if necessary.
Unlike the Carry Out stage (which produces results), this stage focuses on checking, validating,
↪  and reflecting on those results.
## Components
1. STRATEGY-VALIDATION
        [Definition] The process of verifying whether the chosen strategy or assumptions were
        ↪  valid.
        [Role]
        - Confirms that the reasoning approach is appropriate
        - Ensures that conclusions follow logically from assumptions
        [Example]
        - Assumption: The sequence is arithmetic → Check differences: 2, 4, 6, 8 → Validation:
        ↪  **Differences are constant → correct assumption**
2. ERROR-DETECTION
        [Definition] Identifying mistakes in reasoning, calculation, or logical steps.
        [Role]
        - Prevents incorrect conclusions
        - Ensures internal consistency
        [Example]
        - Computed: 10 - 3 = 6 → **Error detected: Incorrect arithmetic → should be 7**
3. DIMENSIONAL-ANALYSIS
  [Definition] Checking whether the units or dimensions of the result are consistent with the
  ↪  required quantity.
  [Role]
  - Ensures physical or logical correctness
  - Detects hidden inconsistencies
  [Example]
        - Speed = distance / time → **Unit check: meters / seconds → valid**
4. EXTREME-CASE-TESTING
        [Definition] Testing whether the solution holds under extreme or boundary conditions.
        [Role]
        - Reveals hidden flaws
        - Tests robustness of the solution
        [Example]
        - f(x) = x / (x + 1) → Test x = 0 → f(0) = 0 → valid → Test x →  → f(x) → 1 → consistent
5. FINAL-ANSWER
        [Definition] The process of clearly presenting the final result derived from the
        ↪  problem-solving process in the required format.
        [Role]
        - Communicates the solution
        - Ensures clarity and completeness
        - Matches the required output format
        [Example]
        - Therefore, the values are **x = 3 and y = 7**.
```

```
----------------------------------

Tagging Schema (use EXACT labels below):

EXTRACTION
SYMBOLIZATION.DIRECT-MAPPING
SYMBOLIZATION.REFRAMING
SYMBOLIZATION.ABSTRACTION
STATE-SPACE-DEFINITION
PATTERN-MATCHING
STRUCTURAL-ANALYSIS.SYMMETRY
STRUCTURAL-ANALYSIS.INVARIANCE
STRUCTURAL-ANALYSIS.DECOMPOSITION
IDEALIZATION
HYPOTHESIS-FORMATION
RETRIEVAL.RECALL
RETRIEVAL.MATCHING
INFERENCE-FLOW.CHAINING.DEDUCTION
INFERENCE-FLOW.CHAINING.INDUCTION
INFERENCE-FLOW.CHAINING.ANALOGY
INFERENCE-FLOW.BRANCHING
EXECUTION.INSTANTIATION
EXECUTION.OPERATION.ALGEBRAIC-MANIPULATION
EXECUTION.OPERATION.ARITHMETIC-COMPUTATION
EXECUTION.OPERATION.LOGICAL-EVALUATION
REPRESENTATION-CONSTRUCTION
PATTERN-EXTRACTION
STRATEGY-VALIDATION
ERROR-DETECTION
DIMENSIONAL-ANALYSIS
EXTREME-CASE-TESTING
FINAL-ANSWER
----------------------------------

Output ONLY a valid JSON array. No explanation, no markdown, no other text.

Format:
[
  {"start": 3, "end": 7, "tag": "TAG_NAME"},
  {"start": 12, "end": 15, "tag": "TAG_NAME"}
]

Where start and end are inclusive word indices (the §N numbers).
A span can cross multiple lines if the reasoning continues.

When end  start > 20 (span covers more than 20 words), you MUST add a "reason" field
explaining in one sentence why it is one continuous reasoning act:
  {"start": 45, "end": 120, "tag": "TAG_NAME", "reason": "..."}

Rules:
- start and end must be §N indices that actually appear in the text
- Do not overlap spans
- Do not tag filler words or transitions unless they are part of a reasoning span
- A single span may cover multiple lines if it is one continuous reasoning act
- If end  start > 20, include a "reason" field (one sentence, concrete)
- If unsure, leave untagged rather than guess
- Tag ONLY the key reasoning content, not filler words.
- Do NOT wrap entire sentences unless absolutely necessary.
- Focus on:
  - problem type (pattern)
  - structural insights
  - key equations or transformations
  - branching conditions
  - final conclusions
- Output only the JSON array, nothing else
```

994

```
----------------------------------

Example 1:

[Question]
What is 2 + 3?

[Solution]
§0 We §1 add §2 2 §3 and §4 3 §5 to §6 get §7 5.

[Output]
[
  {"start": 2, "end": 7, "tag": "EXECUTION.OPERATION.ARITHMETIC-COMPUTATION"},
  {"start": 7, "end": 7, "tag": "FINAL-ANSWER"}
]

----------------------------------

Example 2:

[Question]
A store has 3 types of products. Type A costs $5 each, Type B costs $8 each, Type C costs $12 each.
↪  If a customer buys 4 of type A, 3 of type B, and 2 of type C, what is the total cost?

[Solution]
§0 To §1 find §2 the §3 total §4 cost, §5 I §6 need §7 to §8 calculate §9 each §10 type §11
↪  separately §12 and §13 add §14 them §15 together.
§16 Type §17 A: §18 4 §19 items §20 × §21 $5 §22 = §23 $20.
§24 Type §25 B: §26 3 §27 items §28 × §29 $8 §30 = §31 $24.
§32 Type §33 C: §34 2 §35 items §36 × §37 $12 §38 = §39 $24.
§40 Now §41 I §42 combine §43 these §44 results: §45 $20 §46 + §47 $24 §48 + §49 $24 §50 = §51 $68.
§52 Therefore, §53 the §54 total §55 cost §56 is §57 $68.

[Output]
[
  {"start": 0, "end": 15, "tag": "STRUCTURAL-ANALYSIS.DECOMPOSITION"},
  {"start": 18, "end": 23, "tag": "EXECUTION.OPERATION.ARITHMETIC-COMPUTATION"},
  {"start": 26, "end": 31, "tag": "EXECUTION.OPERATION.ARITHMETIC-COMPUTATION"},
  {"start": 34, "end": 39, "tag": "EXECUTION.OPERATION.ARITHMETIC-COMPUTATION"},
  {"start": 45, "end": 51, "tag": "EXECUTION.OPERATION.ARITHMETIC-COMPUTATION"},
  {"start": 52, "end": 57, "tag": "FINAL-ANSWER"}
]

----------------------------------

Example 3 (long span requiring "reason"):

[Question]
For each integer n from 1 to 9, check whether n is divisible by 3 and sum the digit counts of
↪  those that are.

[Solution]
§0 I §1 will §2 check §3 each §4 integer §5 from §6 1 §7 to §8 9. §9 n=1: §10 digit §11 sum §12 1,
↪  §13 not §14 divisible §15 by §16 3. §17 n=2: §18 digit §19 sum §20 2, §21 not §22 divisible §23
↪  by §24 3. §25 n=3: §26 digit §27 sum §28 3, §29 divisible §30 by §31 3, §32 count §33 it. §34
↪  n=4: §35 digit §36 sum §37 4, §38 not §39 divisible §40 by §41 3. §42 n=5: §43 digit §44 sum
↪  §45 5, §46 not §47 divisible §48 by §49 3. §50 n=6: §51 digit §52 sum §53 6, §54 divisible §55
↪  by §56 3, §57 count §58 it. §59 n=7: §60 digit §61 sum §62 7, §63 not §64 divisible §65 by §66
↪  3. §67 n=8: §68 digit §69 sum §70 8, §71 not §72 divisible §73 by §74 3. §75 n=9: §76 digit
↪  §77 sum §78 9, §79 divisible §80 by §81 3, §82 count §83 it. §84 Total §85 qualifying: §86 3.

[Output]
[
  {"start": 0, "end": 83, "tag": "EXECUTION.OPERATION.LOGICAL-EVALUATION", "reason": "repeated
  ↪  application of the same digit-sum divisibility check across all nine consecutive integers"},
  {"start": 84, "end": 86, "tag": "FINAL-ANSWER"}
]
```

```
----------------------------------
Example 4(Other examples)
§0 Flip §1 a §2 coin §3 three §4 times §5 → §6 This §7 is §8 a §9 probability §10 + §11
↪  combinatorics §12 problem
[{"start": 6, "end": 12, "tag": "PATTERN-MATCHING"}]

§0 Find §1 the §2 shortest §3 path §4 between §5 points §6 → §7 This §8 becomes §9 a §10 graph §11
↪  / §12 optimization §13 problem
[{"start": 7, "end": 13, "tag": "PATTERN-MATCHING"}]

§0 The §1 sum §2 of §3 two §4 numbers §5 is §6 10 §7 → §8 x §9 + §10 y §11 = §12 10
[{"start": 8, "end": 12, "tag": "SYMBOLIZATION.DIRECT-MAPPING"}]

§0 x² §1 = §2 4 §3 → §4 x §5 = §6 2 §7 or §8 x §9 = §10 -2
[{"start": 4, "end": 10, "tag": "INFERENCE-FLOW.BRANCHING"}]
%\end{lstlisting}
```

User Prompt

```
%\begin{lstlisting}[style=promptstyle]
[Question]
{question}

---
[Solution]
{reasoning}
% \end{lstlisting}
```