HIDDEN MARKOV MODELING OF REASONING DYNAMICS IN LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reasoning in language models involves both explicit steps in the generated text and implicit structural shifts in hidden states, yet their joint dynamics remain largely underexplored. We propose a Hierarchical Hidden Markov Model (HHMM) that captures these two dimensions: semantic roles and latent depth regimes. This framework models how reasoning evolves through semantic stages and how the depth of computation shifts across the network. By linking *what* function a step serves to *where* it arises in the network, our approach provides a unified lens for both understanding reasoning dynamics and offering insights into steering strategies. Our analysis reveals consistent patterns: successful reasoning trajectories follow stable semantic paths and align with well-formed structural anchors, whereas failures are characterized by hesitation loops and unstable depth transitions. We further validate our findings by applying step-aware intervention: we derive steering vectors from the transition matrices that encourage trajectories to follow the paths associated with correct reasoning. Across multiple open-source reasoning models, these targeted nudges consistently convert failing runs into correct ones without increasing output length.

1 INTRODUCTION

Reasoning models have demonstrated strong capabilities in mathematics, scientific reasoning, and deliberative problem solving (Jaech et al., 2024; Chen et al., 2025b). Yet fundamental questions remain unresolved: *when models perform reasoning, how do they integrate both **explicit reasoning** expressed in generated text and the **implicit reasoning** reflected in hidden computations to work together in solving problems?*

Recent studies suggest that reasoning effectiveness depends not only on the correctness of individual steps but also on their structural roles, with a few "anchor" steps disproportionately shaping outcomes (Bogdan et al., 2025). Mechanistic studies show that reasoning does not occur uniformly, but is carried out by specific architectural modules (Cabannes et al., 2024; Dutta et al., 2024).

To systematically characterize reasoning in language models, we introduce a *Hierarchical Hidden Markov Model (HHMM)* that integrates explicit and implicit reasoning into a single framework. At the *top level*, the model is a Markov chain over *explicit reasoning categories*, capturing semantic transitions across generated "thinking" tokens. At the *bottom level*, conditioned on the top-level category, the model is a hidden Markov chain over *implicit reasoning regimes*, which represent how computation is allocated across layers within each stage. These two levels are not isolated; rather, they interact hierarchically: semantic roles guide depth allocation, while depth patterns shape semantic progress. This dual perspective allows us to analyze reasoning as trajectories across both meaning and structure. The HHMM thus provides a principled probabilistic framework that connects *what* function a step serves with *where* it arises in the network, offering a unified lens for understanding reasoning dynamics and designing effective interventions.

Explicit and implicit reasoning interact in systematic ways. At the *semantic level*, models branch early into two pathways: a "think-first" route that proceeds step by step through analysis, and a "commit-early" route that jumps directly to an answer. Failures often occur when trajectories stall in verification loops rather than progressing to closure. At the *structural level*, reasoning categories exhibit distinctive depth profiles: *final-answer* steps trigger shifts mainly in the upper layers, whereas *analysis* and *verification* introduce earlier shifts that alter model behavior. Successful trajectories

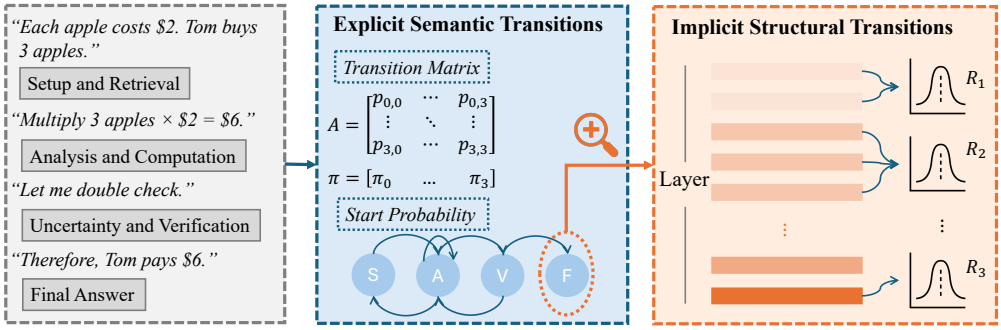


Figure 1: Overview of the Hierarchical Hidden Markov Model (HHMM) framework. At the *top level*, explicit reasoning is modeled as semantic transitions across tagged reasoning steps. At the *bottom level*, implicit reasoning is captured through latent regimes that characterize depth allocation across layers. Together, this hierarchical structure links what function a step serves with where it is realized in the network, enabling systematic analysis of reasoning trajectories.

align these two dimensions: stable semantic progress supported by well-formed late-layer anchors. In contrast, unsuccessful ones diverge, inserting unstable extra shifts or omitting expected ones.

Taken together, these findings suggest that reasoning outcomes are not determined by isolated steps, but by the overall transition flow. Correct predictions emerge when trajectories follow stable and well-anchored transitions, while failures arise when transitions deviate into loops or unstable paths.

We validate this view through step-aware interventions. By deriving steering vectors from the transition matrices, we nudge trajectories toward the transitions associated with correct reasoning and apply them to model hidden states. Across multiple open-source reasoning models, these targeted interventions consistently rescue failing runs without increasing output length, confirming that semantic and structural insights jointly shape model behavior.

Our contributions. We introduce HHMM, a structured probabilistic framework that integrates semantic reasoning stages with structural depth regimes. Using this framework, we show that successful reasoning trajectories follow stable semantic paths and align with well-formed structural anchors, whereas failures are characterized by hesitation loops and unstable depth transitions. Finally, we leverage HHMM to construct edge-conditioned steering vectors, demonstrating that targeted interventions at reasoning boundaries can causally correct errors.

2 Methodology

To systematically characterize reasoning in language models, we introduce a *Hierarchical Hidden Markov Model (HHMM)* that organizes reasoning into two levels (Figure 1). At the *top level*, the model is a Markov chain over *explicit reasoning categories*, capturing semantic transitions across generated "thinking" tokens. At the *bottom level*, conditioned on the top-level category, the model is a hidden Markov chain over *implicit reasoning regimes*, representing how computation is distributed across layers within each stage. These two levels are not isolated; rather, they interact hierarchically, with semantic roles guiding depth allocation and depth patterns shaping semantic progress. The HHMM thus provides a principled probabilistic framework that unifies these complementary dimensions, aligning with annotated reasoning steps while uncovering latent structural patterns that distinguish successful from failing trajectories.

We segment each generated solution into reasoning steps $s_1, \ldots, s_T$ using blank-line delimiters. In parallel, we extract the hidden representations of the first token of each step, forming $H \in \mathbb{R}^{(L+1)\times T\times d}$, where L is the number of layers and d the hidden dimension, with $h_{t,\ell} = H_{\ell,t,:}$ denoting the hidden state of layer ℓ for step s_t. Before modelling, these hidden vectors are standardized to zero mean and unit variance, and then projected into a d_{pca}-dimensional subspace using PCA ($d_{pca} = 64$). This preprocessing both stabilizes Gaussian estimation and removes redundant correlations across hidden dimensions, ensuring that the latent regimes capture meaningful structural variations in depth allocation. Implementation details in Appendix A and B.

2.1 TOP LEVEL: EXPLICIT REASONING TRANSITIONS

At the top level, we model the sequence of reasoning steps as semantic transitions, capturing how the function of each step evolves across the trajectory. For each step s_t, we classify its reasoning function using the same model that produced the solution, prompted with the predefined tag set $\mathcal{C} = \{\text{final\_answer}, \text{setup\_and\_retrieval}, \text{analysis\_and\_computation}, \text{uncertainty\_and\_verification}\}$, yielding a label $c_t \in \mathcal{C}$ (details in Appendix A). This self-classification approach allows us to capture what the model itself believes it is doing at each step. For a trajectory comprising T steps, each step is assigned a category label $c_t \in 1, \ldots, C$. These categorical assignments define a Markov chain over the sequence of reasoning steps,

$$p(c_{1:T}) \;=\; \pi^{\text{top}}_{c_1} \prod_{t=2}^{T} A^{\text{top}}_{c_{t-1},c_t}$$

where π^{top} is the start distribution and A^{top} is the transition matrix.

2.2 BOTTOM LEVEL: IMPLICIT REASONING TRANSITIONS

At the bottom level, conditioned on a semantic category c_t, the model introduces latent regimes that characterize how computation is distributed across layers, capturing depth allocation patterns that vary with the function of the current reasoning step. Each layer index ℓ within step t is associated with a latent regime variable $h_{t,\ell} \in \{1, \ldots, K\}$, where K denotes the number of regimes. Intuitively, each regime represents a characteristic way the model distributes computation across layers within a given semantic category. Remaining in the same regime indicates stable, consistent processing, while switching regimes highlights points where the layer-wise behavior changes abruptly, often signaling a shift in how the model is reasoning internally. Every category induces its own stochastic process over regimes, allowing the model to capture distinct depth allocation patterns for different reasoning stages:

$$p(h_{t,0:L} \mid c_t) \;=\; \pi^{(c)}_{h_{t,0}} \prod_{\ell=1}^{L} A^{(c)}_{h_{t,\ell-1},h_{t,\ell}},$$

with Gaussian emissions,

$$p(h_{t,\ell} \mid h_{t,\ell} = k,\, c_t) \;=\; \mathcal{N}\big(h_{t,\ell};\, \mu^{(c)}_k,\, \text{diag}(\sigma^{2(c)}_k)\big).$$

Here, $\pi^{(c)}$ and $A^{(c)}$ are the initial and transition distributions over regimes for category c, while $\mu^{(c)}_k$ and $\sigma^{2(c)}_k$ define the emission parameters for regime k. This structure allows the model to discover latent depth allocation patterns within each semantic stage.

The HHMM framework provides a structured lens on reasoning dynamics: semantic transitions capture what stages of reasoning unfold and how they progress, while structural regimes capture how computation is internally organized within each stage. By comparing models trained on correct versus incorrect trajectories, we uncover where and how reasoning paths diverge.

3 EXPERIMENTAL SETUP

Model. We evaluate 4 open-source reasoning models: Qwen3-1.7B (Team, 2025), Bespoke-Stratos-7B (Labs, 2025), OpenThinker-7B (Guha et al., 2025), and Llama-3.1-Nemotron-Nano-4B-v1.1 (Bercovich et al., 2025).

Dataset. Our experiments use: MATH-500 (Lightman et al., 2023), GPQA-Diamond (Rein et al., 2024), WebInstruct-Verified (Ma et al., 2025), and AIME-2024 (HuggingFaceH4, 2024).

Computing. Experiments were run on NVIDIA H100 and H200 GPUs.

4 SEMANTIC REASONING DYNAMICS

To understand how language models reason, three questions naturally arise: *Where do reasoning trajectories tend to flow? What separates successful reasoning paths from failures? And how do different models differ in their trajectories?*

Q1. Where do reasoning trajectories flow?

Reasoning unfolds in structured patterns rather than randomly. As shown in Table 1, trajectories display three recurring motifs: they often stabilize within analysis and answer emission stages, branch at setup into two alternative pathways, and occasionally take short verification detours that loop back to analysis.

i) **Attractor stages.** *Analysis_and_computation* and *final_answer* act as attractors with strong self-loops (0.390 and 0.395), far higher than *setup* (0.255) or *verification* (0.206). Once trajectories enter these states, they rarely leave.

ii) **Balanced branching.** *Setup_and_retrieval* functions as a fork, with nearly equal probability of moving to *analysis_and_computation* (0.317) or jumping directly to *final_answer* (0.317). This balance reflects two styles: a deliberative "think first" pathway versus a shortcut "commit early" pathway.

Table 1: Transition probabilities highlight three motifs: strong self-loops at *final* and *analysis*, balanced exits from *setup* to *analysis* vs. *final*, and a transient *verification* that mostly feeds back to *analysis*.

	Final	Setup	Analysis	Verify
Final	0.395	0.180	0.295	0.130
Setup	0.317	0.255	0.317	0.111
Analysis	0.299	0.179	0.390	0.132
Verify	0.277	0.200	0.318	0.206

iii) **Checkpoint loop.** *Uncertainty_and_verification* is used as a transient detour. Transitions from *analysis_and_computation* into *Uncertainty_and_verification* are infrequent (0.132), while returns to *analysis_and_computation* are common (0.318). Rather than serving as a destination, verification acts as a loop-back checkpoint.

Q2. What separates success from failure?

Successful runs generally begin in *setup_and_retrieval* and then reach *final_answer* efficiently. This happens either directly (setup→final = 0.329) or indirectly via *analysis_and_computation* (setup→analysis = 0.309, followed by analysis→final = 0.323). Once in *final_answer*, correct trajectories strongly stabilize (final→final = 0.414).

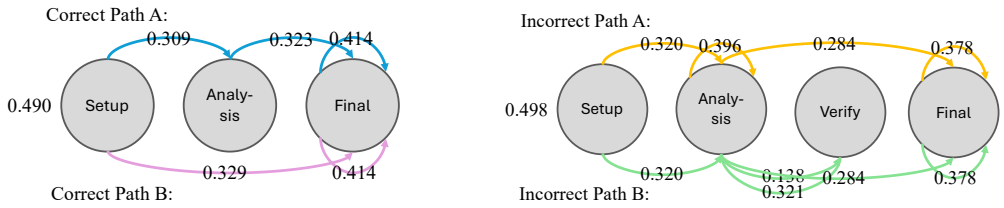
Table 2: Comparison of start distributions and transition probabilities for correct vs. incorrect trajectories. Correct runs favor two efficient entries from *setup* and stabilize in *final/analysis*, while incorrect runs dwell in *analysis* and cycle with *verification*, with weaker closure in *final*.

	Correct					**Incorrect**				
	Start Prob	Transition Matrix				Start Prob	Transition Matrix			
		Final	Setup	Analysis	Verify		Final	Setup	Analysis	Verify
Final	0.201	0.414	0.179	0.286	0.122	0.186	0.378	0.179	0.309	0.135
Setup	0.492	0.329	0.257	0.309	0.105	0.498	0.308	0.257	0.320	0.115
Analysis	0.215	0.323	0.173	0.382	0.122	0.218	0.284	0.183	0.396	0.138
Verify	0.092	0.299	0.192	0.316	0.193	0.098	0.263	0.205	0.321	0.211

In contrast, unsuccessful runs also start in *setup_and_retrieval* but become stuck in *analysis_and_computation*, reinforced by a strong self-loop (0.396) and repeated cycling with *uncertainty_and_verification* (analysis→verification = 0.138, verification→analysis = 0.321). Their stabilization in *final_answer* is weaker (final→final = 0.378), showing indecision and incomplete closure. Table 2 and Figure 2 summarize these differences.

Q3: Why do models diverge in their reasoning paths?

Divergence reflects both architectural biases and inherited generation styles (Table 3).



(a) Reasoning path for correct prediction: (Path A) setup→final, (Path B) setup→analysis→final

(b) Reasoning path for incorrect prediction: (Path A) setup→analysis(loop)→final, (Path B) setup→analysis→verify→analysis→final.

Figure 2: Contrasting reasoning paths: successful trajectories commit and close; unsuccessful ones hesitate—looping within *analysis*/*verification* before reaching *final*.

Architectural bias. Qwen-based models (Qwen3, Stratos, Openthinker) exhibit a *deliberation-oriented* style: starting in *setup*, lingering in *analysis*, and occasionally looping through *verification*. By contrast, Nemotron (LLaMA-based) shows a *direct-commit* bias, with high probability of setup→final transitions (0.671) and strong reinforcement once in *final* (final→final = 0.665). This reveals a sharp divide between extended deliberation versus confident commitment.

Generation style. Training lineage also shapes characteristic "thinking profiles." Stratos and Openthinker, both distilled from R1 with Qwen2.5-7B-Instruct, converge on a balanced pattern: setup splits between analysis (0.343) and final (0.236), analysis maintains a moderate self-loop (0.373), and verification is engaged but limited (0.234). Qwen3, built on the same Qwen family but not follow R1 style, pushes deliberation even further: setup dominates the start distribution (0.836) and analysis sustains a strong self-loop (0.522). In this way, Stratos and Openthinker inherit balanced dynamics from distillation, while Qwen3 diverges toward a more deliberation-heavy style—highlighting how lineage both transmits and reshapes reasoning patterns.

Table 3: Comparison of key transition statistics across models. Nemotron shows a *direct-commit* bias (high setup→final and strong final self-loop), Qwen3 amplifies *deliberation* (dominant setup start and persistent analysis), while Stratos and Openthinker converge on a more *balanced* profile.

Model	Start in Setup	Setup→Final	Setup→Analysis	Analysis Self-loop	Final Self-loop	Verify Self-loop
Nemotron	0.153	**0.671**	0.178	0.300	**0.665**	0.090
Qwen3	**0.836**	0.132	**0.409**	**0.522**	0.391	0.273
Stratos	0.490	0.231	0.335	0.367	0.262	0.228
Openthinker	0.496	0.236	0.343	0.373	0.260	0.234

5 Structural Reasoning Dynamics

So far we have examined reasoning through the lens of surface-level semantics. We now turn to the *internal structure* of reasoning: how models carve up their depth into distinct processing phases. To analyze this, we treat the HHMM-decoded **phase boundaries** as empirical objects that mark where one depth regime ends and another begins. Our HHMM is trained with $C = 4$ top-level categories (final_answer, setup_and_retrieval, analysis_and_computation, uncertainty_and_verification) and $K = 7$ bottom-level regimes per category, where K allows the model to represent fine-grained micro-states within each semantic stage. A methodological challenge is that architectures differ in depth: Stratos, Openthinker and Qwen3 have 29 layers, whereas Nemotron has 33. To compare them fairly, we normalize all layers to a *percent-depth scale* $[0, 99]$, mapping the embedding projection to 0.0 and the final transformer block to 99. Boundaries are then represented as normalized *endpoints*, and similarity is measured using Jaccard overlap between endpoint sets.

Q1. Where do models consistently place boundaries?

To highlight stable anchors, we first normalize boundaries to percent depth and apply a two-stage voting scheme: within each model, a boundary must appear in at least 30% of runs to be kept; across

models, it is included in the *consensus set* if at least two out of four models agree. This procedure filters out noise and reveals boundaries that recur reliably across architectures.

Table 4 lists the consensus splits for each category, and Figure 3 shows cross-model endpoint similarity. The heatmap quantifies overlap in endpoint placement, showing strong alignment between Openthinker and Stratos, partial alignment with Qwen, and divergence from Nemotron. Across all categories, boundaries cluster into four regions: **early** (0–37), **mid** (38–71), **upper** (72–89), and **top** (90–99). The exact subdivisions differ by category:

(i) *Final_answer* has a long early-to-mid phase (0–71) followed by successive cuts in the upper and top bands.

(ii) *Analysis_and_computation* inserts finer granularity, with multiple cuts in the mid and upper regions.

(iii) *Setup_and_retrieval* mirrors *final_answer* but with a distinct upper-band pattern.

(iv)*Uncertainty_and_verification* is the only category with an early split (0–0.37), before converging on the same upper and top anchors.

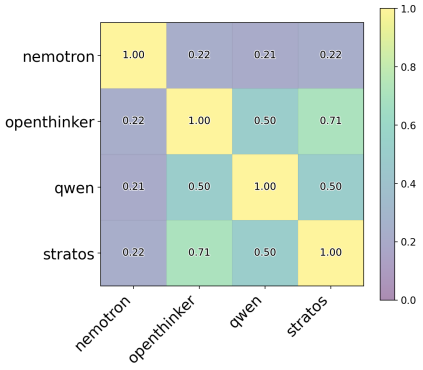


Figure 3: Cross-model similarity quantifies overlap in endpoint placement.

Together, these results show that reasoning "endgames" are highly conserved in the upper layers, while the early layers differentiate categories: *analysis* and *verification* engage early boundaries, whereas *setup* and *final_answer* remain quiescent until later stages.

Table 4: Consensus boundaries (percent depth) across four models. All categories converge on similar upper and top anchors, while analysis and verification also introduce early splits.

Category	Majority Splits (percent depth)
analysis_and_computation	0–37, 38–71, 72–75, 76–82, 83–85, 86–89, 90–92, 93–96, 97–99
final_answer	0–71, 72–78, 79–82, 83–85, 86–89, 90–92, 93–96, 97–99
setup_and_retrieval	0–71, 72–75, 76–78, 79–82, 83–89, 90–92, 93–96, 97–99
uncertainty_and_verification	0–37, 38–68, 69–71, 72–78, 79–82, 83–85, 86–89, 90–92, 93–96, 97–99

Q2. Do correct and incorrect runs use the same boundaries?

Consensus anchors also reveal systematic differences between correct and incorrect trajectories. Table 5 summarizes these contrasts. Correct runs converge on a compact set of canonical boundaries, producing repeatable anchor points across models. Incorrect runs, by contrast, deviate in two ways: (i) they insert extra boundaries, fragmenting mid-to-upper regions into smaller segments, or (ii) they collapse structure by omitting established anchors. These deviations are especially pronounced in *analysis_and_computation*, where incorrect runs frequently add mid-layer boundaries. For instance, Nemotron expands from a stable 5-split structure in correct cases to 7 splits in incorrect cases (see Appendix E). By contrast, *final_answer* anchors remain highly conserved: correct and incorrect trajectories alike converge on nearly identical late-layer closure, differing only in granularity.

Overall, consensus analysis shows that incorrect reasoning is not random noise but reflects systematic disruptions to canonical depth segmentation: either over-segmentation, which fragments processing, or under-segmentation, which fails to sustain necessary stages.

6 Semantic Transitions as Functional Control Points

In the previous sections, we established that the structure of reasoning paths matters. In this section, we use *steering* as a validation tool: by deriving vectors from the semantic transition matrix and applying them as interventions at reasoning boundaries, we test whether nudging trajectories along these directions can causally alter model performance.

Table 5: Extremum stability summary from consensus anchors: correct vs. incorrect trajectories differ in canonical split points.

Category	Avg. splits (Correct)	Avg. splits (Incorrect)	Δ Splits	Jaccard Overlap (shared boundaries)
analysis_and_computation	5–9	7–8	-1–+2	Low (11–62%)
final_answer	3–10	7–10	+0–+4	Medium (14–89%)
setup_and_retrieval	6–8	5–9	-2–+2	Medium (33–80%)
uncertainty_and_verification	4–9	5–9	-2–+1	Medium (40–71%)

6.1 Build Steering Vectors

Edge-conditioned Displacement Vectors. From §2, we know that correct and incorrect trajectories follow distinct semantic transition patterns. To capture and target these divergences, we construct **edge-conditioned displacement vectors**. Let $\Phi : \mathbb{R}^d \to \mathbb{R}^k$ denote the preprocessing mentioned in §2. For each step s_t, we take the final-layer hidden state $h_{t,L} \in \mathbb{R}^d$ and map it to $z_t = \Phi(h_{t,L}) \in \mathbb{R}^k$. For every adjacent category pair $(a, b) \in \mathcal{C} \times \mathcal{C}$ induced by (c_t, c_{t+1}), we compute displacement vectors $d_t = z_{t+1} - z_t$, and separate these displacements into two sets: those from correct trajectories and those from incorrect ones. For correct trajectories, we obtain an edge-specific vector by averaging all d_t along edge (a, b). For incorrect trajectories, instead of using (a, b) directly, we construct a source-conditioned baseline: the average of displacements originating from the same source category a, but across all incorrect outgoing edges other than (a, b):

$$\mu^{\mathsf{corr}}_{a\to b} = \frac{1}{N^{\mathsf{corr}}_{a\to b}} \sum_{(t:\, y=\mathsf{corr},\, c_t=a,\, c_{t+1}=b)} d_t, \mu^{\mathsf{srcneg}}_{a\to b} = \frac{1}{N^{\mathsf{inc}}_{a} - N^{\mathsf{inc}}_{a\to b}} \sum_{t:\, y=\mathsf{inc},\, c_t=a,\, c_{t+1}\neq b} d_t.$$

The $\Delta_{a\to b}$ is computed as $\mu^{\mathsf{corr}}_{a\to b} - \mu^{\mathsf{srcneg}}_{a\to b} \in \mathbb{R}^k$, and then normalized. To perform steering in the model's hidden space, we inverse preprocess Φ and map $v_{a\to b} = \Phi^{-1}(\Delta_{a\to b}) \in \mathbb{R}^d$. See Algorithm 1.

Edge Selection from Divergent Transitions. A key question is which transitions should be targeted for steering. Since correct and incorrect trajectories differ most strongly on specific category transitions, we rank edges by the difference in their transition probabilities, measured as $A^{\mathsf{corr}} - A^{\mathsf{inc}}$. We then consider two strategies:

i) *Max-Divergence Edge.* Select the single edge with the largest positive difference. For example, correct trajectories frequently move from *setup_and_retrieval* to *analysis_and_computation*, whereas incorrect ones are more likely to loop within *setup_and_retrieval*, or to jump prematurely to *final_answer* or *uncertainty_and_verification*.

ii) *Multi-Edge Ensemble.* Select the top three edges with the largest positive differences, capturing a broader set of corrective transitions.

In both cases, the steering vector $v_{a\to b}$ serves as a directional signal: it promotes the desired correct transition $(a \to b)$ while counteracting the competing incorrect alternatives.

Steering Protocol. During generation, we apply $v_{a\to b}$ only at step boundaries, detected by the model producing a blank line delimiter. At the first token after such a boundary, we add $v_{a\to b}$ directly to the hidden state at the final transformer layer, just before the unembedding to logits. $\alpha > 0$ controls the intervention strength, and the update is $h \leftarrow h + \alpha\, v_{a\to b}$.

We introduce two steering methods. i) In hard steering, the intervention vector is deterministically taken from the max-divergence edge, i.e., always applying $v_{a\to b}$ from the single highest-ranked transition. ii) In soft steering, we relax this choice by introducing stochasticity: from the bank of the top-3 divergent edges $\{(a_k \to b_k,\, p_k)\}$, we normalize the scores so that $\sum_k p_k = 1$, then sample an index $k \sim \mathrm{Categorical}(p_1, \ldots, p_3)$ once per batch and apply the corresponding vector $v_{a_k\to b_k}$. This makes steering probabilistic, letting the intervention randomly select among the most divergent candidates instead of always committing to one.

Algorithm 1: Edge-conditioned steering vector construction

Input: Final-layer states $\{h_{t,L}\}$, categories $\{c_t\}$, correctness labels y, preprocessing Φ, edge set E.
Output: Steering vectors $v_{a\to b} \in \mathbb{R}^d$.
for *each step t in each trajectory* **do**
  $z_t = \Phi(h_{t,L}),\ z_{t+1} = \Phi(h_{t+1,L}),\ d_t = z_{t+1} - z_t$;
  if $y =$ *corr* **then**
    add d_t to bucket $(a{=}c_t, b{=}c_{t+1})$ for correct
  else
    add d_t to bucket (a, b) for incorrect and to source bucket a
for *each edge* $(a, b) \in E$ **do**
  $\mu^{\text{corr}}_{a\to b}$ = mean of correct displacements on (a, b);
  $\mu^{\text{srcneg}}_{a\to b}$ = mean of incorrect displacements from a excluding (a, b);
  $\Delta_{a\to b}$ = normalize$(\mu^{\text{corr}}_{a\to b} - \mu^{\text{srcneg}}_{a\to b})$;
  $v_{a\to b} = \Phi^{-1}(\Delta_{a\to b})$;

6.2 Intervention

We inject edge-conditioned vectors into the model's hidden states at reasoning boundaries and evaluate their impact (Table 6). Three steering methods are compared: (i) *edge-agnostic steering*, which uses a single global vector derived from the overall correct–incorrect contrast and applies it uniformly across all transitions, and (ii) the two *edge-conditioned steering* methods introduced in the steering protocol. Motivated by our structural analysis, where the final layer consistently forms its own regime and *final-answer* steps consolidate in the upper layers, we target the last layer as the natural site for intervention. The left half of Table 6 reports the correction rate, i.e., the fraction of originally incorrect predictions that are corrected after steering. The right half reports the change in output length, measured as token counts before and after steering. If edge-conditioned steering achieves higher correction rates than the edge-agnostic baseline, this provides causal evidence that choosing the right reasoning path improves model performance.

Table 6: Correction rate (↑; fraction of originally incorrect predictions corrected after steering) and false-token count (↓; token counts of originally incorrect predictions), averaged over 3 independent runs. Top values are highlighted.

Model	Dataset	Steering Correction Rate ↑			Tokens ↓		
		Edge-Agnostic	Soft Steering	Hard Steering	Baseline	Soft Steering	Hard Steering
Llama-3.1-Nemotron-Nano-4B-v1.1	MATH-500	0.1081	0.1219	0.1383	1986	1949	1941
	WebInstruct-Verified	0.0969	0.1181	0.0966	4330	4173	4185
	GPQA-Diamond	0.1423	0.1653	0.1630	4833	4845	4796
OpenThinker-7B	MATH-500	0.2324	0.2443	0.2311	1899	1758	1785
	WebInstruct-Verified	0.2367	0.2408	0.2094	3474	3303	3442
	GPQA-Diamond	0.1858	0.1619	0.2126	4340	4155	4204
Qwen3-1.7B	MATH-500	0.1198	0.1312	0.1175	1998	1983	1982
	WebInstruct-Verified	0.1091	0.1295	0.1062	3926	3889	3878
	GPQA-Diamond	0.0709	0.0792	0.0774	4526	4567	4492
Bespoke-Stratos-7B	MATH-500	0.2832	0.2962	0.2747	1688	1537	1547
	WebInstruct-Verified	0.1679	0.1949	0.2065	1532	1424	1507
	GPQA-Diamond	0.2431	0.2682	0.1900	2695	2729	2587
	Average	0.1664	**0.1793**	0.1686	3102	3026	3029

Targeted interventions improve accuracy without added tokens. Steering at semantic edges consistently improves correction rates while leaving output length unaffected. On average, correction improves from 0.1664 to 0.1793 under soft steering, while false-token counts drop from 3102 to 3026. These results demonstrate that *reasoning paths directly affect model performance*: targeted interventions can rescue failures by encouraging trajectories toward the transitions associated with correct reasoning.

Final-answer transitions are central to success. To identify which transitions matter most, we analyze the most frequent top-1 edges across models, summarized in Table 7. At the global level, the dominant transition is *uncertainty_and_verification* $\rightarrow$ *final_answer*. Nemotron and Stratos strongly favor this same transition, Qwen relies most on *analysis_and_computation* $\rightarrow$ *final_answer*, while OpenThinker frequently reinforces *final_answer* $\rightarrow$ *final_answer*.

This concentration shows that models succeed through a set of decisive transitions rather than distributing progress evenly across the reasoning graph. Typical patterns include resolving uncertainty by committing to a final answer, concluding analysis with a decision, or reinforcing an reached answer for consistency. The steering experiments confirm: reasoning in LMs is structured by a latent reasoning graph, where specific edges serve as functional decision points that govern success or failure.

Table 7: Most common top-1 semantic transitions across models and datasets.

Scope	Top-1 Edge
Global	uncertainty_and_verification $\rightarrow$ final_answer
Nemotron	uncertainty_and_verification $\rightarrow$ final_answer
OpenThinker	final_answer $\rightarrow$ final_answer
Qwen	analysis_and_computation $\rightarrow$ final_answer
Stratos	uncertainty_and_verification $\rightarrow$ final_answer

7 Related Work

Recent studies highlight that reasoning models can exhibit unintended or deceptive behaviors, underscoring the need for a deeper mechanistic understanding (Baker et al., 2025). Several works examine which aspects of CoT steps matter. Madaan & Yazdanbakhsh (2022) disentangle the roles of textual content and structural patterns, Wang et al. (2022) show that performance gains often persist even with flawed step content, and Bogdan et al. (2025) find that a small set of "anchor" steps disproportionately shape final outcomes.

Mechanistic analyses probe the internal processes behind reasoning. Cabannes et al. (2024) and Dutta et al. (2024) show how architectural components enable stepwise reasoning. Latent multi-hop phenomena are revealed by Yang et al. (2024b) and Shalev et al. (2024), while Venhoff et al. (2025) demonstrate that steering vectors can capture functional directions in hidden space. Information-theoretic perspectives provide a complementary lens: Ton et al. (2024) analyze CoT dynamics through entropy and mutual information, while Punjwani & Heck (2025) explore how neural network weights encode and constrain reasoning capacity. Beyond analysis, several works pursue interventions. Chen et al. (2025a) introduce training-free latent steering to suppress over-reflection, while Venhoff et al. (2025) show that representation-level modifications can modulate reasoning style. Wang et al. (2025b) propose efficient post-training refinement of latent reasoning, enabling reasoning improvements without full retraining. Reasoning efficiency survey (Sui et al., 2025) catalog methods to mitigate "overthinking," and token-level analyses (Wang et al., 2025a) identify sparse high-entropy tokens as critical intervention points. Two concurrent works explicitly model state-aware dynamics of reasoning. Wu et al. (2025) frame CoT as a latent-state MDP, training a transition policy with RL to improve reasoning exploration. Yu et al. (2025) cluster final-layer embeddings of steps and construct a Markov chain to visualize reasoning motifs. Our work differs by introducing a *hierarchical HMM* that integrates explicit semantic roles with hidden layer regimes. This design ties *what* function a step serves to *where* it arises in the network, and further enables *HHMM-informed steering vectors* that rescue failing trajectories through structured interventions.

8 Conclusion

We introduced a Hierarchical Hidden Markov Model (HHMM) that integrates semantic reasoning roles with latent depth regimes, linking *what* a step does to *where* it arises in the network. This dual perspective reveals consistent dynamics: successful trajectories follow stable semantic paths anchored by well-formed structural boundaries, while failures arise from hesitation loops, unstable shifts, or missing anchors. Beyond providing an explanatory lens, HHMM also enables control—steering vectors derived from transition patterns can nudge trajectories toward successful reasoning paths and reliably rescue failing runs without increasing output length. Taken together, these results position HHMM as both a framework for understanding reasoning and a foundation for structured interventions that improve the robustness and reliability of reasoning models.

Ethics Statement. This work adheres to the Code of Ethics. Our experiments use only open-source models and publicly available datasets under their respective open licenses, with no involvement of human subjects or sensitive data. We identify no foreseeable ethical risks.

Reproducibility Statement. We ensure reproducibility by providing experimental and implementation details in Section 3 and Appendices A–E. Full results with statistical significance are in Appendices C–E, and anonymous source code is included as supplementary material.

REFERENCES

Bowen Baker, Joost Huizinga, Leo Gao, Zehao Dou, Melody Y Guan, Aleksander Madry, Wojciech Zaremba, Jakub Pachocki, and David Farhi. Monitoring reasoning models for misbehavior and the risks of promoting obfuscation. *arXiv preprint arXiv:2503.11926*, 2025.

Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, et al. Llama-nemotron: Efficient reasoning models. *arXiv preprint arXiv:2505.00949*, 2025.

Paul C Bogdan, Uzay Macar, Neel Nanda, and Arthur Conmy. Thought anchors: Which llm reasoning steps matter? *arXiv preprint arXiv:2506.19143*, 2025.

Vivien Cabannes, Charles Arnal, Wassim Bouaziz, Xingyu Yang, Francois Charton, and Julia Kempe. Iteration head: A mechanistic study of chain-of-thought. *Advances in Neural Information Processing Systems*, 37:109101–109122, 2024.

Runjin Chen, Zhenyu Zhang, Junyuan Hong, Souvik Kundu, and Zhangyang Wang. Seal: Steerable reasoning calibration of large language models for free. *arXiv preprint arXiv:2504.07986*, 2025a.

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Somani, Peter Hase, Misha Wagner, Fabien Roger, et al. Reasoning models don't always say what they think. *arXiv preprint arXiv:2505.05410*, 2025b.

Subhabrata Dutta, Joykirat Singh, Soumen Chakrabarti, and Tanmoy Chakraborty. How to think step-by-step: A mechanistic understanding of chain-of-thought reasoning. *arXiv preprint arXiv:2402.18312*, 2024.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Benjamin Feuer, Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, Wanjia Zhao, John Yang, Shreyas Pimpalgaonkar, Kartik Sharma, Charlie Cheng-Jie Ji, Yichuan Deng, Sarah Pratt, Vivek Ramanujan, Jon Saad-Falcon, Jeffrey Li, Achal Dave, Alon Albalak, Kushal Arora, Blake Wulfe, Chinmay Hegde, Greg Durrett, Sewoong Oh, Mohit Bansal, Saadia Gabriel, Aditya Grover, Kai-Wei Chang, Vaishaal Shankar, Aaron Gokaslan, Mike A. Merrill, Tatsunori Hashimoto, Yejin Choi, Jenia Jitsev, Reinhard Heckel, Maheswaran Sathiamoorthy, Alexandros G. Dimakis, and Ludwig Schmidt. Openthoughts: Data recipes for reasoning models, 2025. URL https://arxiv.org/abs/2506.04178.

HuggingFaceH4. Aime_2024 dataset. https://huggingface.co/datasets/HuggingFaceH4/aime_2024, 2024.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Bespoke Labs. Bespoke-stratos: The unreasonable effectiveness of reasoning distillation. https://www.bespokelabs.ai/blog/bespoke-stratos-the-unreasonable-effectiveness-of-reasoning-distillation, 2025.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Xueguang Ma, Qian Liu, Dongfu Jiang, Ge Zhang, Zejun Ma, and Wenhu Chen. General-Reasoner: Advancing llm reasoning across all domains. *arXiv:2505.14652*, 2025. URL https://arxiv.org/abs/2505.14652.

Aman Madaan and Amir Yazdanbakhsh. Text and patterns: For effective chain of thought, it takes two to tango. *arXiv preprint arXiv:2209.07686*, 2022.

Meta. Llama 3.1 8b instruct. https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct, 2024. Accessed: 2025-09-24.

Saif Punjwani and Larry Heck. Weight-of-thought reasoning: Exploring neural network weights for enhanced llm reasoning. *ArXiv*, abs/2504.10646, 2025. URL https://api.semanticscholar.org/CorpusID:277787278.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

Yuval Shalev, Amir Feder, and Ariel Goldstein. Distributional reasoning in llms: Parallel reasoning processes in multi-hop reasoning. *arXiv preprint arXiv:2406.13858*, 2024.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, et al. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*, 2025.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Jean-François Ton, Muhammad Faaiz Taufiq, and Yang Liu. Understanding chain-of-thought in llms through information theory. *ArXiv*, abs/2411.11984, 2024. URL https://api.semanticscholar.org/CorpusID:274141313.

Constantin Venhoff, Iván Arcuschin, Philip Torr, Arthur Conmy, and Neel Nanda. Understanding reasoning in thinking language models via steering vectors. *arXiv preprint arXiv:2506.18167*, 2025.

Boshi Wang, Sewon Min, Xiang Deng, Jiaming Shen, You Wu, Luke Zettlemoyer, and Huan Sun. Towards understanding chain-of-thought prompting: An empirical study of what matters. *arXiv preprint arXiv:2212.10001*, 2022.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, et al. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*, 2025a.

Xinyuan Wang, Dongjie Wang, Wangyang Ying, Haoyue Bai, Nanxu Gong, Sixun Dong, Kunpeng Liu, and Yanjie Fu. Efficient post-training refinement of latent reasoning in large language models. *ArXiv*, abs/2506.08552, 2025b. URL https://api.semanticscholar.org/CorpusID:279260460.

Junda Wu, Yuxin Xiong, Xintong Li, Zhengmian Hu, Tong Yu, Rui Wang, Xiang Chen, Jingbo Shang, and Julian McAuley. Ctrls: Chain-of-thought reasoning via latent state-transition. *arXiv preprint arXiv:2507.08182*, 2025.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024a.

Sohee Yang, Elena Gribovskaya, Nora Kassner, Mor Geva, and Sebastian Riedel. Do large language models latently perform multi-hop reasoning? *arXiv preprint arXiv:2402.16837*, 2024b.

Sheldon Yu, Yuxin Xiong, Junda Wu, Xintong Li, Tong Yu, Xiang Chen, Ritwik Sinha, Jingbo Shang, and Julian McAuley. Explainable chain-of-thought reasoning: An empirical analysis on state-aware reasoning dynamics. *arXiv preprint arXiv:2509.00190*, 2025.

A Data Preprocessing and Experimental Setup

A.1 Models

We evaluate five open-source reasoning models, each paired with a classification model where applicable. All models are run with bfloat16 precision unless otherwise specified. Generation models use standard reasoning hyperparameters, while classification models are configured with do_sample = False. Licenses are listed for reproducibility.

Qwen3-1.7B (Team, 2025) was used for both generation and classification, with "thinking mode" enabled for generation and disabled for classification. The parameters were: temperature = 0.6, top_p = 1.0, top_k = 20, and min_p = 0; classification was run deterministically with do_sample = False. License: Apache 2.0.

Bespoke-Stratos-7B (Labs, 2025) was paired with *Qwen2.5-7B-Instruct* (Yang et al., 2024a) for classification. The configuration used temperature = 0.6, top_p = 0.95, and deterministic classification (do_sample = False). License: Apache 2.0 (both models).

OpenThinker-7B (Guha et al., 2025) was paired with *Qwen2.5-7B-Instruct* (Yang et al., 2024a), using the same settings (temperature = 0.6, top_p = 0.95, and do_sample = False for classification). License: Apache 2.0 (both models).

Llama-3.1-Nemotron-Nano-4B-v1.1 (Bercovich et al., 2025) was used for both generation and classification. "Thinking mode" was enabled for generation and disabled for classification, with parameters temperature = 0.6, top_p = 0.95, and deterministic classification (do_sample = False). License: NVIDIA Open Model License.

Besides above models, LLaMA-3.1-8B-Instruct (Meta, 2024) are used only for WebInstruct-Verified, with Meta LLaMA 3.1 License.

A.2 Datasets

Our experiments span 4 benchmarks. Splits, sizes, maximum token cutoffs and licenses are shown in Table 8. AIME-2024 is excluded from steering analysis due to its small size and difficulty.

Table 8: Datasets used in experiments.

Dataset	Split	Train Size	Test Size	Max Tokens	License
MATH-500 (Lightman et al., 2023)	test	250	250	2000	MIT
GPQA-Diamond (Rein et al., 2024)	test	100	98	5000	Apache 2.0
WebInstruct-Verified (Ma et al., 2025)	test	144	144	5000	Apache 2.0
AIME-2024 (HuggingFaceH4, 2024)	train	15	–	32768	Apache 2.0

For WebInstruct-Verified, we additionally filter by: answer_types as Float, Multiple Choice, Integer, Percentage and difficulties as Primary, Junior High, Senior High.

A.3 Prompting

Generation. - **Multiple-choice tasks (GPQA)** use:

```
You are answering a multiple-choice question.
Options are labeled A, B, C, and D.
Think step-by-step and show your reasoning.
At the very end, output ONE line exactly in this format:
Final Answer: \boxed{A}
```

- **Open-ended tasks (MATH-500, AIME-2024, WebInstruct)** use:

```
Answer the following question step-by-step.
At the very end, output exactly one line formatted as:
Final Answer: \boxed{...}
```

Exception: For Stratos on MATH-500, no prompt is used, as this improved performance.

Classification. For sentence-level semantic tagging, all classification models use:

```
You are an expert in reasoning analysis.
Classify the function of each sentence into one of the following tags:
1. final_answer
2. setup_and_retrieval
3. analysis_and_computation
4. uncertainty_and_verification
```

A.4 Preprocessing Pipeline

We segment each generated solution into reasoning steps using blank-line delimiters, and for every step we store the hidden representation of its first token across all layers as the step-level hidden state. Per-step reasoning sequences are constructed and anchored to semantic labels.

A.5 Computing

All experiments were conducted with NVIDIA H100 and H200 GPUs.

B HHMM Configurations

B.1 Training

Data Loading and Filtering Classification labels and step-level hidden states (extracted during Appendix A) are loaded. Depending on the configuration, training uses either all samples, only correct samples, or only incorrect samples.

Feature Preprocessing Step embeddings are standardized with a StandardScaler, then reduced via PCA to a target dimension of 64 (d_{pca} = 64). PCA is run with the solver option svd_solver="full".

HHMM Fitting The HHMM is trained with four fixed top-level categories ($C = 4$) aligned to semantic anchors, and seven bottom-level regimes ($K = 7$) per category. Unless otherwise specified, training runs for 10 EM iterations (n_iter = 10).

Anchored Top-Level Categories The HHMM is trained in anchored mode, where top-level categories are fixed by semantic labels. Each reasoning step is assigned one of four canonical categories: final_answer, setup_and_retrieval, analysis_and_computation, uncertainty_and_verification. These labels are required for every sequence and are coerced into integer IDs (0–3). The top-level start and transition probabilities are estimated directly from observed label sequences.

Bottom-Level HMMs For each category, a bottom-level HMM with $K = 7$ regimes is trained to model the sequence of hidden states across layers. Emissions are diagonal Gaussians with parameters (μ_k, σ_k^2), initialized from a pooled sample of step embeddings. Variances are lower-bounded to 10^{-3} for stability.

Expectation–Maximization (EM) Training proceeds via EM for $n_{\text{iter}} = 10$ iterations: (i) E-step: For each step, given its fixed category, the bottom HMM runs forward–backward to compute regime posteriors and sufficient statistics. (ii) M-step: Top-level parameters (start and transition) are updated by normalized counts over fixed labels; bottom-level parameters are re-estimated from sufficient statistics. Average per-step log-likelihood is monitored across iterations.

B.2 Decoding

Inputs and Preprocessing Restore Anchored decoding operates on two inputs: (1) preprocessed records containing step-level hidden states and labels, and (2) a HHMM checkpoint that stores model

parameters and preprocessing statistics. The original StandardScaler and PCA are reconstructed by directly restoring their learned attributes, rather than refitting.

Anchored Decoding Bottom-level states are decoded with Viterbi under the fixed top-level category path. Anchored decoding constrains the top-level HHMM categories using provided labels and returns best_regimes_per_step for each sequence, which is the fine-grained bottom-level regime path assigned by model.

Consensus Layer Ranges To summarize regime allocation across layers, we derive *consensus ranges* for each top category:

1. **Frequency pooling.** For each category c, build frequency tensors $\mathrm{reg\_freq}[c] \in \mathbb{N}^{L\times K}$, where entry (i, r) counts how often regime r is assigned at layer i.
2. **Dominant regime selection.** At each layer i, choose the regime with the largest relative frequency: $r^* = \arg\max_r \ \mathrm{reg\_freq}[c]_{i,r} \big/ \sum_{r'} \mathrm{reg\_freq}[c]_{i,r'}$.
3. **Range consolidation.** Merge consecutive layers that share the same dominant regime into one interval to obtain compact ranges.

Table 9: Example.

Regime	Layers
5	0–6
3	7–18
1	19–22
6	23–25
0	26–27
6	28

C Steering Vector Construction and Intervention

C.1 Steering Vector Construction

We provide implementation details for the steering vector construction, which derives steering vectors from HHMM transition matrix and hidden states. These vectors capture directional displacements in representation space that distinguish correct from incorrect reasoning paths.

Input. The build script consumes three inputs: (1) model hidden state, (2) semantic level transition matrix, and (3) trained preprocessing models containing scaling statistics and optional PCA components.

Preprocessing. Each hidden state H is standardized using a fitted StandardScaler, then reduced with PCA to dimension $d_{pca} = 64$. This yields a single step embedding $v_t \in \mathbb{R}^{d_{pca}}$ per reasoning step.

Edge statistics. For each category transition $(a \to b)$ in correct runs, we compute the average displacement $\Delta_{a,b} = \mathbb{E}[v_{t+1} - v_t]$. For incorrect runs, we maintain both edge-conditional and source-marginal statistics, then construct a *source-conditioned negative* baseline. Subtracting this baseline from correct displacements yields contrastive vectors $\Delta_{a,b}$ that emphasize features predictive of successful reasoning.

Baselines. A global baseline is computed by averaging hidden vectors across all steps of correct vs. incorrect runs, independent of edge structure. This yields a steering direction v_{global} for comparison.

Soft edge weighting. When choose short steering, a json file will be used to specify a combination of multiple edges with normalized weights, allowing multi-edge steering vectors.

Output. Per-edge steering vectors, baseline vector, and soft steering configuration with normalized probability.

C.2 Steering at Inference Time

This section describes how steering vectors are applied during generation.

Goal. Given a hidden-space steering vector $v \in \mathbb{R}^d$ (inverted back to the model's hidden width from the PCA/scaler space), we add a small vector $\alpha * v$ at chosen transformer layers during decoding to encourage trajectories associated with successful reasoning.

steering Vectors Steering operates by adding precomputed vectors in the model's hidden space. In the hard-steering mode, we load one vector; instead in the soft-steering mode, we instead build a small bucket of edge-specific vectors, each associated with a probability weight. At inference, one vector from this bucket is selected by random sampling, and the chosen vector is applied as an intervention to steer the hidden states.

Injection Mechanism (Forward Hooks) During generation, we register lightweight forward hooks on the selected blocks. Let $h \in \mathbb{R}^{B\times T\times d}$ be the hidden state for last layer for a batch of size B. When the gate fires (Sec. C.2), we update the last token representation for each active batch slot b:

$$h_{b,T-1,:} \leftarrow h_{b,T-1,:} + \alpha\, v.$$

Step-Aware Gating To make steering *step-aware*, we attach a logits-processor that arms a per-batch gate when the generated text ends with a blank line. The next token after arming receives the steering update and the gate resets.

Table 10: Correction rate (↑; fraction of originally incorrect predictions corrected after steering, with standard deviations) and false-token count (↓; token counts of originally incorrect predictions, with standard deviations), average and std over 3 independent runs.

Model	Dataset	Edge-Agnostic	Soft Steering		Hard Steering		Baseline	Soft Steering		Hard Steering		#False
		Corr.	Corr.	Std	Corr.	Std	Tokens	Tokens	Std	Tokens	Std	
Llama-3.1-Nemotron-Nano-4B-v1.1	MATH-500	0.1081	0.1219	0.04	0.1383	0.04	1986	1949	14.71	1941	8.86	120
	WebInstruct-Verified	0.0969	0.1181	0.04	0.0966	0.03	4330	4173	27.78	4185	65.18	69
	GPQA-Diamond	0.1423	0.1653	0.03	0.1630	0.09	4833	4845	54.56	4796	77.11	106
OpenThinker-7B	MATH-500	0.2324	0.2443	0.07	0.2311	0.02	1899	1758	10.06	1785	29.00	80
	WebInstruct-Verified	0.2367	0.2408	0.03	0.2094	0.04	3474	3303	283.88	3442	35.01	96
	GPQA-Diamond	0.1858	0.1619	0.08	0.2126	0.04	4340	4155	88.74	4204	125.16	72
Qwen3-1.7B	MATH-500	0.1198	0.1312	0.01	0.1175	0.02	1998	1983	7.65	1982	10.09	100
	WebInstruct-Verified	0.1091	0.1295	0.03	0.1062	0.01	3926	3889	97.23	3878	54.68	125
	GPQA-Diamond	0.0709	0.0792	0.04	0.0774	0.02	4526	4567	55.88	4492	51.80	81
Bespoke-Stratos-7B	MATH-500	0.2832	0.2962	0.02	0.2747	0.06	1688	1537	44.92	1547	18.25	59
	WebInstruct-Verified	0.1679	0.1949	0.03	0.2065	0.04	1532	1424	139.98	1507	46.69	87
	GPQA-Diamond	0.2431	0.2682	0.12	0.1900	0.07	2695	2729	252.48	2587	101.64	76
	Average	0.1664	**0.1793**		0.1686		3102	3026		3029		

D Semantic HHMM Transition Analysis

Goal. We summarize the *top–level* (semantic) dynamics of our HHMM by averaging start–state probabilities and transition matrices across runs, and by contrasting correct vs. incorrect trajectories.

Aggregation. For any slice $\mathcal{D}$, we compute

$$\bar{T} = \text{mean}_{r\in\mathcal{D}}\, T^{(r)} \in \mathbb{R}^{C\times C}, \qquad \bar{S} = \text{mean}_{r\in\mathcal{D}}\, S^{(r)} \in \mathbb{R}^{C},$$

where $T^{(r)}$ and $S^{(r)}$ are the per–run transition matrix and start distribution. We additionally report a difference view $\Delta T = \bar{T}_{\text{correct}} - \bar{T}_{\text{incorrect}},\ \Delta S = \bar{S}_{\text{correct}} - \bar{S}_{\text{incorrect}}$.

Reading guide (key takeaways). (i) Starts. Most sequences start in setup_and_retrieval (≈0.49); correct runs show a slightly higher chance to start in final_answer (+1.5pp).

(ii) Progress toward final_answer. Correct runs are consistently *more likely* to move into final_answer from any state (row–wise ΔT to final_answer is +2.1–3.9pp), with the largest lift from analysis_and_computation (+3.9pp).

(iii) Stalls and detours. Incorrect runs show relatively higher self–loops or detours into analysis_and_computation/uncertainty_and_verification (negative entries in the to analysis_and_computation/uncertainty_and_verification columns of ΔT).

E Structural HHMM Transition Analysis

This section describes the procedure for aggregating layer "endpoints" (structural boundaries) across models, subsets.

E.1 Percent-depth bins.

Each absolute endpoint $e \in [0..L]$ is mapped to a bin $k \in [0..B]$ (with $B = 99$):

$$k = \text{round}\big(\tfrac{e}{L} \cdot B\big)\,.$$

Boundary bins $\{0, B\}$ are always included.

$$MODEL\_CAPS = \{stratos = 29,\ openthinker = 29,\ qwen = 29,\ nemotron = 33\}.$$

E.2 Within-Model Consensus Voting

We compute per-file endpoint sets in $[0..B]$ and then vote across runs. An endpoint $u \notin \{0, B\}$ is kept if

$$\frac{\#\{\text{runs containing } u\}}{n_f} \ \geq\ ENDPOINT\_THR,$$

where n_f is the number of unique runs. We set $ENDPOINT\_THR = 0.3$.

E.3 Across-Model Voting

After within-model consensus, endpoints are aggregated across models. Let $\mathcal{E}_m$ be the voted endpoints for model m. Then keep bins present in at least two models (out of four total).

E.4 Consensus Stability Summary (Correct vs. Incorrect)

From the per-subset consensus, we compute stability statistics for each category:

(i) **Splits:** $n_{\text{splits}} = |\mathcal{E}| - 1$ per subset, model, category, aggregated as averages and min–max across models.

(ii) Δ **Splits:** per model, $\Delta = n^{(\text{inc})}_{\text{splits}} - n^{(\text{cor})}_{\text{splits}}$; reported as the extremal range $[\min \Delta, \max \Delta]$.

(iii) **Anchor overlap (Jaccard):** anchors are interior endpoints $\mathcal{A} = \mathcal{E} \setminus \{0, B\}$. For each model:

$$J = \frac{|\mathcal{A}_{\text{cor}} \cap \mathcal{A}_{\text{inc}}|}{|\mathcal{A}_{\text{cor}} \cup \mathcal{A}_{\text{inc}}|}.$$

We report per-category averages, min–max, and bucket averages as Low (< 0.5), Medium ($[0.5, 0.7)$), or High (≥ 0.7).

E.5 Heatmaps

To quantify similarity between models, we construct heatmaps of endpoint overlap. For each pair of models (m_1, m_2), we compute the histogram overlap of their endpoint distributions in percent-depth space.

Table 11: Mean top–level transition matrices ($\bar{T}$). Rows are sources; columns are destinations.

	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
ALL				
final_answer	0.395	0.180	0.295	0.130
setup_and_retrieval	0.317	0.255	0.317	0.111
analysis_and_computation	0.299	0.179	0.390	0.132
uncertainty_and_verification	0.277	0.200	0.318	0.206
CORRECT				
final_answer	0.414	0.179	0.286	0.122
setup_and_retrieval	0.329	0.257	0.309	0.105
analysis_and_computation	0.323	0.173	0.382	0.122
uncertainty_and_verification	0.299	0.192	0.316	0.193
INCORRECT				
final_answer	0.378	0.179	0.309	0.135
setup_and_retrieval	0.308	0.257	0.320	0.115
analysis_and_computation	0.284	0.183	0.396	0.138
uncertainty_and_verification	0.263	0.205	0.321	0.211

Table 12: Difference matrix $\Delta T = \bar{T}_{\text{correct}} - \bar{T}_{\text{incorrect}}$. Positive values indicate transitions that are *more likely* in correct runs.

	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
final_answer	**+0.036**	+0.000	-0.023	-0.013
setup_and_retrieval	**+0.021**	-0.000	-0.010	-0.011
analysis_and_computation	**+0.039**	-0.009	-0.013	-0.016
uncertainty_and_verification	**+0.036**	-0.013	-0.005	-0.018

F USE OF LARGE LANGUAGE MODELS.

Large language models (LLMs) were used solely as assistive tools for proofreading and improving clarity of writing.

Table 13: Mean start distributions ($\bar{S}$) and difference $\Delta S = \bar{S}_{\text{correct}} - \bar{S}_{\text{incorrect}}$.

	ALL	CORRECT	INCORRECT
final_answer	0.191	0.201	0.186
setup_and_retrieval	0.494	0.492	0.498
analysis_and_computation	0.220	0.215	0.218
uncertainty_and_verification	0.095	0.092	0.098
Difference ΔS (Correct − Incorrect): [+0.015, -0.006, -0.003, -0.006]			

Table 14: AIME-2024 — Mean top–level transition matrices ($\bar{T}$). Rows are sources; columns are destinations.

	final_answer	setup_and_retrieval	analysis_and_computation	uncertainty_and_verification
ALL				
final_answer	0.377	0.177	0.329	0.116
setup_and_retrieval	0.305	0.251	0.344	0.099
analysis_and_computation	0.303	0.174	0.401	0.123
uncertainty_and_verification	0.265	0.197	0.339	0.199
CORRECT				
final_answer	0.399	0.172	0.321	0.108
setup_and_retrieval	0.327	0.237	0.338	0.097
analysis_and_computation	0.339	0.163	0.389	0.108
uncertainty_and_verification	0.271	0.199	0.347	0.183
INCORRECT				
final_answer	0.363	0.181	0.339	0.117
setup_and_retrieval	0.294	0.261	0.344	0.101
analysis_and_computation	0.288	0.178	0.405	0.129
uncertainty_and_verification	0.263	0.199	0.338	0.200

Table 15: AIME-2024 — Difference matrix $\Delta T = \bar{T}_{\text{correct}} - \bar{T}_{\text{incorrect}}$. Positive values are more likely in correct runs.

	final_answer	setup_and_retrieval	analysis_and_computation	uncertainty_and_verification
final_answer	**+0.036**	-0.009	-0.018	-0.009
setup_and_retrieval	**+0.033**	-0.024	-0.006	-0.004
analysis_and_computation	**+0.051**	-0.015	-0.015	-0.021
uncertainty_and_verification	+0.008	+0.000	+0.009	-0.017

Table 16: AIME-2024 — Mean start distributions ($\bar{S}$) and $\Delta S = \bar{S}_{\text{correct}} - \bar{S}_{\text{incorrect}}$.

	ALL	CORRECT	INCORRECT
final_answer	0.228	0.231	0.217
setup_and_retrieval	0.506	0.527	0.486
analysis_and_computation	0.150	0.115	0.181
uncertainty_and_verification	0.117	0.126	0.116
Difference ΔS (Correct − Incorrect): [+0.014, +0.041, -0.066, +0.011]			

Table 17: GPQA-Diamond — Mean top–level transition matrices ($\bar{T}$). Rows are sources; columns are destinations.

	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
ALL				
final_answer	0.400	0.191	0.252	0.157
setup_and_retrieval	0.313	0.278	0.258	0.150
analysis_and_computation	0.280	0.202	0.337	0.181
uncertainty_and_verification	0.276	0.200	0.274	0.249
CORRECT				
final_answer	0.412	0.186	0.252	0.149
setup_and_retrieval	0.315	0.300	0.246	0.139
analysis_and_computation	0.287	0.200	0.344	0.169
uncertainty_and_verification	0.298	0.183	0.273	0.245
INCORRECT				
final_answer	0.400	0.191	0.251	0.158
setup_and_retrieval	0.313	0.273	0.261	0.154
analysis_and_computation	0.276	0.203	0.337	0.184
uncertainty_and_verification	0.268	0.207	0.274	0.251

Table 18: GPQA-Diamond — Difference matrix $\Delta T = \bar{T}_{\text{correct}} - \bar{T}_{\text{incorrect}}$. Positive values are more likely in correct runs.

	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
final_answer	**+0.012**	-0.005	+0.001	-0.009
setup_and_retrieval	+0.003	+0.027	-0.015	-0.014
analysis_and_computation	+0.011	-0.003	+0.008	-0.015
uncertainty_and_verification	+0.030	-0.023	-0.001	-0.006

Table 19: GPQA-Diamond — Mean start distributions ($\bar{S}$) and $\Delta S = \bar{S}_{\text{correct}} - \bar{S}_{\text{incorrect}}$.

	ALL	**CORRECT**	**INCORRECT**
final_answer	0.201	0.229	0.188
setup_and_retrieval	0.499	0.499	0.502
analysis_and_computation	0.207	0.191	0.212
uncertainty_and_verification	0.093	0.082	0.098
Difference ΔS (Correct − Incorrect): [+0.041, -0.004, -0.021, -0.016]			

Table 20: MATH-500 — Mean top–level transition matrices ($\bar{T}$). Rows are sources; columns are destinations.

	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
ALL				
final_answer	0.389	0.196	0.317	0.097
setup_and_retrieval	0.309	0.264	0.353	0.074
analysis_and_computation	0.304	0.183	0.422	0.091
uncertainty_and_verification	0.272	0.225	0.353	0.150
CORRECT				
final_answer	0.414	0.194	0.301	0.092
setup_and_retrieval	0.332	0.253	0.347	0.068
analysis_and_computation	0.337	0.173	0.408	0.082
uncertainty_and_verification	0.305	0.214	0.347	0.134
INCORRECT				
final_answer	0.349	0.190	0.353	0.107
setup_and_retrieval	0.284	0.273	0.360	0.083
analysis_and_computation	0.268	0.192	0.437	0.103
uncertainty_and_verification	0.241	0.238	0.365	0.157

Table 21: MATH-500 — Difference matrix $\Delta T = \bar{T}_{\text{correct}} - \bar{T}_{\text{incorrect}}$. Positive values are more likely in correct runs.

	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
final_answer	**+0.065**	+0.003	-0.052	-0.015
setup_and_retrieval	+0.048	-0.020	-0.013	-0.015
analysis_and_computation	+0.069	-0.019	-0.029	-0.021
uncertainty_and_verification	+0.064	-0.024	-0.018	-0.023

Table 22: MATH-500 — Mean start distributions ($\bar{S}$) and $\Delta S = \bar{S}_{\text{correct}} - \bar{S}_{\text{incorrect}}$.

	ALL	**CORRECT**	**INCORRECT**
final_answer	0.159	0.155	0.171
setup_and_retrieval	0.446	0.428	0.473
analysis_and_computation	0.310	0.334	0.273
uncertainty_and_verification	0.084	0.083	0.084
Difference ΔS (Correct − Incorrect): [-0.016, -0.045, +0.061, -0.001]			

Table 23: WebInstruct-Verified — Mean top–level transition matrices ($\bar{T}$). Rows are sources; columns are destinations.

	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
ALL				
final_answer	0.412	0.155	0.283	0.149
setup_and_retrieval	0.342	0.227	0.311	0.120
analysis_and_computation	0.310	0.157	0.402	0.131
uncertainty_and_verification	0.293	0.176	0.303	0.228
CORRECT				
final_answer	0.430	0.164	0.269	0.138
setup_and_retrieval	0.342	0.237	0.306	0.115
analysis_and_computation	0.329	0.156	0.388	0.127
uncertainty_and_verification	0.323	0.172	0.297	0.208
INCORRECT				
final_answer	0.400	0.153	0.291	0.156
setup_and_retrieval	0.340	0.222	0.314	0.123
analysis_and_computation	0.304	0.156	0.405	0.135
uncertainty_and_verification	0.281	0.177	0.307	0.235

Table 24: WebInstruct-Verified — Difference matrix $\Delta T = \bar{T}_{\text{correct}} - \bar{T}_{\text{incorrect}}$. Positive values are more likely in correct runs.

	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
final_answer	**+0.030**	+0.010	-0.022	-0.018
setup_and_retrieval	+0.002	+0.015	-0.008	-0.009
analysis_and_computation	+0.024	+0.000	-0.018	-0.007
uncertainty_and_verification	+0.042	-0.005	-0.010	-0.027

Table 25: WebInstruct-Verified — Mean start distributions ($\bar{S}$) and $\Delta S = \bar{S}_{\text{correct}} - \bar{S}_{\text{incorrect}}$.

	ALL	**CORRECT**	**INCORRECT**
final_answer	0.177	0.191	0.169
setup_and_retrieval	0.524	0.513	0.531
analysis_and_computation	0.211	0.220	0.207
uncertainty_and_verification	0.088	0.077	0.093
Difference ΔS (Correct − Incorrect): [+0.022, -0.018, +0.013, -0.017]			

Table 26: Nemotron — Mean top–level transition matrices ($\bar{T}$). Rows are sources; columns are destinations.

	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
ALL				
final_answer	0.665	0.095	0.199	0.041
setup_and_retrieval	0.671	0.119	0.178	0.031
analysis_and_computation	0.548	0.102	0.300	0.050
uncertainty_and_verification	0.549	0.107	0.254	0.090
CORRECT				
final_answer	0.671	0.088	0.192	0.048
setup_and_retrieval	0.678	0.108	0.176	0.038
analysis_and_computation	0.583	0.088	0.277	0.053
uncertainty_and_verification	0.585	0.096	0.238	0.081
INCORRECT				
final_answer	0.659	0.099	0.204	0.038
setup_and_retrieval	0.663	0.125	0.182	0.030
analysis_and_computation	0.531	0.109	0.311	0.049
uncertainty_and_verification	0.528	0.119	0.264	0.090

Table 27: Nemotron — Difference matrix $\Delta T = \bar{T}_{\text{correct}} - \bar{T}_{\text{incorrect}}$. Positive values are more likely in correct runs.

	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
final_answer	**+0.012**	-0.011	-0.011	+0.010
setup_and_retrieval	+0.015	-0.017	-0.006	+0.008
analysis_and_computation	+0.052	-0.021	-0.034	+0.003
uncertainty_and_verification	+0.057	-0.023	-0.025	-0.009

Table 28: Nemotron — Mean start distributions ($\bar{S}$) and $\Delta S = \bar{S}_{\text{correct}} - \bar{S}_{\text{incorrect}}$.

	ALL	**CORRECT**	**INCORRECT**
final_answer	0.386	0.416	0.368
setup_and_retrieval	0.153	0.148	0.148
analysis_and_computation	0.309	0.299	0.317
uncertainty_and_verification	0.152	0.136	0.167
Difference ΔS (Correct − Incorrect): [+0.048, -0.000, -0.017, -0.031]			

Table 29: Openthinker — Mean top–level transition matrices ($\bar{T}$). Rows are sources; columns are destinations.

	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
ALL				
final_answer	0.260	0.219	0.330	0.190
setup_and_retrieval	0.236	0.259	0.343	0.161
analysis_and_computation	0.244	0.206	0.373	0.176
uncertainty_and_verification	0.244	0.212	0.309	0.234
CORRECT				
final_answer	0.285	0.204	0.338	0.174
setup_and_retrieval	0.245	0.262	0.332	0.160
analysis_and_computation	0.267	0.200	0.371	0.162
uncertainty_and_verification	0.252	0.204	0.309	0.235
INCORRECT				
final_answer	0.241	0.230	0.330	0.199
setup_and_retrieval	0.230	0.259	0.349	0.163
analysis_and_computation	0.234	0.209	0.374	0.183
uncertainty_and_verification	0.238	0.220	0.314	0.227

Table 30: Openthinker — Difference matrix $\Delta T = \bar{T}_{\text{correct}} - \bar{T}_{\text{incorrect}}$. Positive values are more likely in correct runs.

	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
final_answer	**+0.044**	-0.026	+0.008	-0.026
setup_and_retrieval	+0.016	+0.004	-0.017	-0.003
analysis_and_computation	+0.033	-0.009	-0.003	-0.021
uncertainty_and_verification	+0.014	-0.016	-0.006	+0.008

Table 31: Openthinker — Mean start distributions ($\bar{S}$) and $\Delta S = \bar{S}_{\text{correct}} - \bar{S}_{\text{incorrect}}$.

	ALL	**CORRECT**	**INCORRECT**
final_answer	0.192	0.205	0.195
setup_and_retrieval	0.496	0.488	0.500
analysis_and_computation	0.198	0.189	0.200
uncertainty_and_verification	0.113	0.119	0.105
Difference ΔS (Correct − Incorrect): [+0.009, -0.012, -0.011, +0.013]			

Table 32: Qwen — Mean top–level transition matrices ($\bar{T}$). Rows are sources; columns are destinations.

	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
ALL				
final_answer	0.391	0.185	0.319	0.105
setup_and_retrieval	0.132	0.371	0.409	0.088
analysis_and_computation	0.174	0.182	0.522	0.123
uncertainty_and_verification	0.083	0.251	0.393	0.273
CORRECT				
final_answer	0.429	0.194	0.280	0.097
setup_and_retrieval	0.155	0.374	0.398	0.073
analysis_and_computation	0.196	0.178	0.517	0.109
uncertainty_and_verification	0.108	0.243	0.407	0.241
INCORRECT				
final_answer	0.359	0.173	0.358	0.110
setup_and_retrieval	0.110	0.385	0.410	0.096
analysis_and_computation	0.149	0.190	0.528	0.132
uncertainty_and_verification	0.068	0.255	0.386	0.292

Table 33: Qwen — Difference matrix $\Delta T = \bar{T}_{\text{correct}} - \bar{T}_{\text{incorrect}}$. Positive values are more likely in correct runs.

	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
final_answer	**+0.070**	+0.021	-0.078	-0.013
setup_and_retrieval	+0.045	-0.010	-0.011	-0.023
analysis_and_computation	+0.047	-0.013	-0.012	-0.023
uncertainty_and_verification	+0.041	-0.011	+0.021	-0.051

Table 34: Qwen — Mean start distributions ($\bar{S}$) and $\Delta S = \bar{S}_{\text{correct}} - \bar{S}_{\text{incorrect}}$.

	ALL	**CORRECT**	**INCORRECT**
final_answer	0.003	0.000	0.003
setup_and_retrieval	0.836	0.832	0.853
analysis_and_computation	0.157	0.164	0.140
uncertainty_and_verification	0.005	0.004	0.004
Difference ΔS (Correct − Incorrect): [-0.003, -0.021, +0.024, -0.000]			

Table 35: Stratos — Mean top–level transition matrices ($\bar{T}$). Rows are sources; columns are destinations.

	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
ALL				
final_answer	0.262	0.219	0.334	0.184
setup_and_retrieval	0.231	0.271	0.335	0.163
analysis_and_computation	0.231	0.225	0.367	0.177
uncertainty_and_verification	0.231	0.227	0.314	0.228
CORRECT				
final_answer	0.269	0.230	0.332	0.168
setup_and_retrieval	0.238	0.283	0.331	0.148
analysis_and_computation	0.245	0.227	0.366	0.163
uncertainty_and_verification	0.251	0.226	0.311	0.212
INCORRECT				
final_answer	0.253	0.214	0.342	0.191
setup_and_retrieval	0.227	0.261	0.339	0.173
analysis_and_computation	0.222	0.221	0.370	0.186
uncertainty_and_verification	0.219	0.227	0.321	0.234

Table 36: Stratos — Difference matrix $\Delta T = \bar{T}_{\text{correct}} - \bar{T}_{\text{incorrect}}$. Positive values are more likely in correct runs.

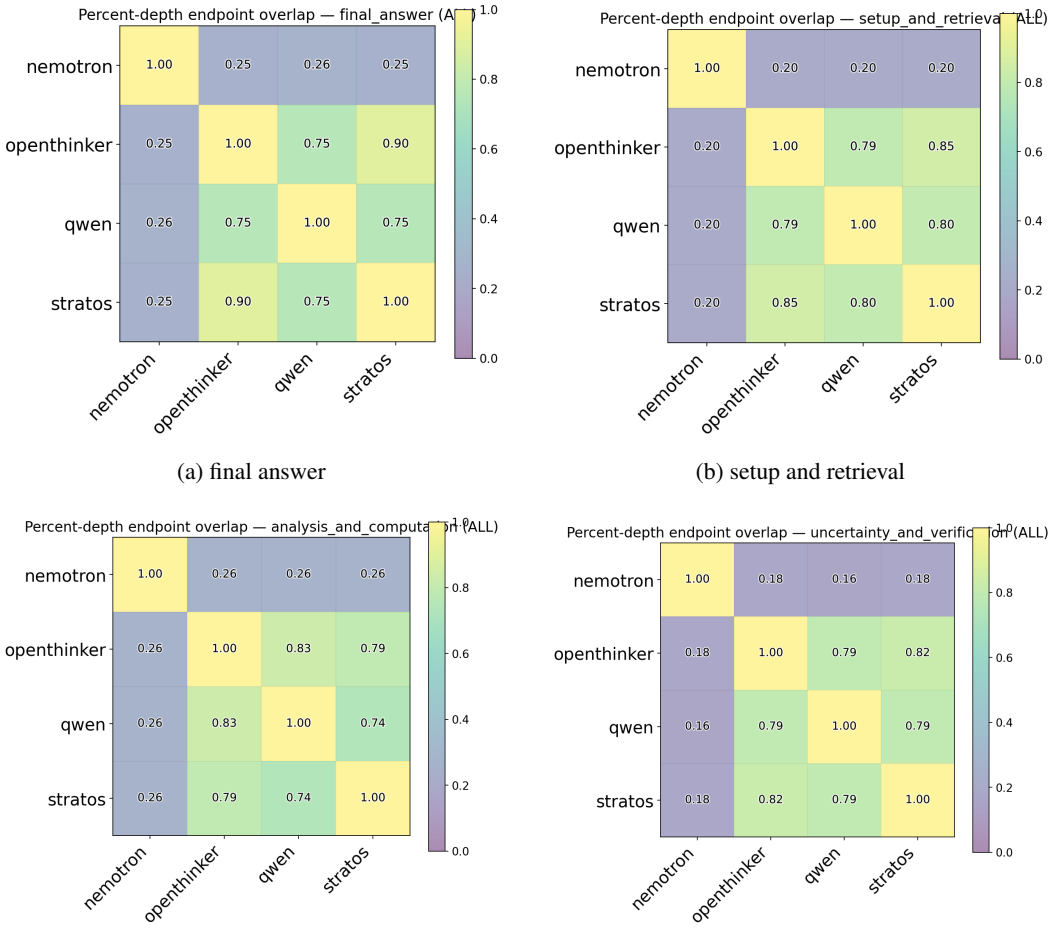
	final_answer	**setup_and_retrieval**	**analysis_and_computation**	**uncertainty_and_verification**
final_answer	**+0.016**	+0.017	-0.010	-0.023
setup_and_retrieval	+0.011	+0.022	-0.008	-0.025
analysis_and_computation	+0.023	+0.006	-0.005	-0.024
uncertainty_and_verification	+0.032	-0.001	-0.010	-0.022

Table 37: Stratos — Mean start distributions ($\bar{S}$) and $\Delta S = \bar{S}_{\text{correct}} - \bar{S}_{\text{incorrect}}$.

	ALL	**CORRECT**	**INCORRECT**
final_answer	0.184	0.185	0.178
setup_and_retrieval	0.490	0.498	0.491
analysis_and_computation	0.214	0.208	0.216
uncertainty_and_verification	0.112	0.109	0.115
Difference ΔS (Correct − Incorrect): [+0.007, +0.007, -0.008, -0.006]			

Table 38: Per-model × category × subset consensus (bins 0..99).

subset	model	category_name	segments_bins_0_99
all	nemotron	analysis_and_computation	0-38—39-41—42-63—64-84—85-93—94-96—97-99
all	openthinker	analysis_and_computation	0-71—72-82—83-85—86-89—90-92—93-96—97-99
all	qwen	analysis_and_computation	0-37—38-61—62-68—69-71—72-75—76-85—86-89—90-96—97-99
all	stratos	analysis_and_computation	0-37—38-71—72-75—76-82—83-85—86-89—90-92—93-96—97-99
all	nemotron	final_answer	0-35—36-38—39-41—42-60—61-81—82-84—85-90—91-96—97-99
all	openthinker	final_answer	0-68—69-71—72-78—79-82—83-85—86-89—90-92—93-96—97-99
all	qwen	final_answer	0-37—38-40—41-61—62-71—72-78—79-85—86-89—90-96—97-99
all	stratos	final_answer	0-47—48-71—72-75—76-82—83-85—86-89—90-92—93-96—97-99
all	nemotron	setup_and_retrieval	0-35—36-41—42-63—64-84—85-90—91-96—97-99
all	openthinker	setup_and_retrieval	0-68—69-71—72-78—79-82—83-89—90-92—93-96—97-99
all	qwen	setup_and_retrieval	0-40—41-61—62-65—66-75—76-78—79-85—86-89—90-96—97-99
all	stratos	setup_and_retrieval	0-37—38-71—72-75—76-82—83-89—90-92—93-96—97-99
all	nemotron	uncertainty_and_verification	0-41—42-96—97-99
all	openthinker	uncertainty_and_verification	0-68—69-71—72-78—79-82—83-85—86-89—90-92—93-96—97-99
all	qwen	uncertainty_and_verification	0-37—38-61—62-68—69-75—76-78—79-82—83-85—86-89—90-96—97-99
all	stratos	uncertainty_and_verification	0-37—38-68—69-71—72-82—83-89—90-92—93-96—97-99
correct	nemotron	analysis_and_computation	0-41—42-66—67-84—85-96—97-99
correct	openthinker	analysis_and_computation	0-37—38-71—72-82—83-85—86-89—90-92—93-96—97-99
correct	qwen	analysis_and_computation	0-37—38-40—41-61—62-75—76-78—79-85—86-89—90-96—97-99
correct	stratos	analysis_and_computation	0-68—69-75—76-78—79-82—83-89—90-92—93-96—97-99
correct	nemotron	final_answer	0-38—39-96—97-99
correct	openthinker	final_answer	0-37—38-68—69-71—72-82—83-85—86-89—90-92—93-96—97-99
correct	qwen	final_answer	0-37—38-40—41-61—62-65—66-71—72-75—76-78—79-89—90-96—97-99
correct	stratos	final_answer	0-71—72-75—76-82—83-89—90-92—93-96—97-99
correct	nemotron	setup_and_retrieval	0-38—39-41—42-63—64-84—85-96—97-99
correct	openthinker	setup_and_retrieval	0-71—72-75—76-82—83-92—93-96—97-99
correct	qwen	setup_and_retrieval	0-40—41-61—62-65—66-75—76-78—79-89—90-96—97-99
correct	stratos	setup_and_retrieval	0-71—72-75—76-82—83-85—86-89—90-92—93-96—97-99
correct	nemotron	uncertainty_and_verification	0-41—42-84—85-96—97-99
correct	openthinker	uncertainty_and_verification	0-71—72-75—76-82—83-89—90-92—93-96—97-99
correct	qwen	uncertainty_and_verification	0-40—41-61—62-65—66-75—76-78—79-85—86-89—90-96—97-99
correct	stratos	uncertainty_and_verification	0-37—38-71—72-82—83-89—90-92—93-96—97-99
incorrect	nemotron	analysis_and_computation	0-35—36-38—39-60—61-81—82-87—88-96—97-99
incorrect	openthinker	analysis_and_computation	0-71—72-78—79-85—86-89—90-92—93-96—97-99
incorrect	qwen	analysis_and_computation	0-37—38-61—62-68—69-75—76-82—83-89—90-96—97-99
incorrect	stratos	analysis_and_computation	0-71—72-75—76-82—83-85—86-89—90-92—93-96—97-99
incorrect	nemotron	final_answer	0-41—42-57—58-78—79-87—88-90—91-96—97-99
incorrect	openthinker	final_answer	0-37—38-68—69-71—72-75—76-82—83-85—86-89—90-92—93-96—97-99
incorrect	qwen	final_answer	0-37—38-40—41-61—62-71—72-75—76-78—79-85—86-89—90-96—97-99
incorrect	stratos	final_answer	0-30—31-68—69-78—79-82—83-85—86-92—93-96—97-99
incorrect	nemotron	setup_and_retrieval	0-38—39-63—64-84—85-96—97-99
incorrect	openthinker	setup_and_retrieval	0-68—69-75—76-78—79-85—86-89—90-92—93-96—97-99
incorrect	qwen	setup_and_retrieval	0-37—38-40—41-61—62-71—72-75—76-85—86-89—90-96—97-99
incorrect	stratos	setup_and_retrieval	0-71—72-82—83-89—90-92—93-96—97-99
incorrect	nemotron	uncertainty_and_verification	0-38—39-63—64-84—85-96—97-99
incorrect	openthinker	uncertainty_and_verification	0-71—72-75—76-85—86-89—90-92—93-96—97-99
incorrect	qwen	uncertainty_and_verification	0-37—38-40—41-61—62-65—66-71—72-82—83-89—90-96—97-99
incorrect	stratos	uncertainty_and_verification	0-71—72-82—83-92—93-96—97-99



(a) final answer

(b) setup and retrieval

(c) analysis and computation

(d) uncertainty and verification

Figure 4: Percent-depth endpoint overlap.