MULTISCALE PREDICTIVE DYNAMICS MAKE MODEL REASONING OBSERVABLE AND CONTROLLABLE

Anonymous authors
Paper under double-blind review

ABSTRACT

Language models routinely perform long multi-step reasoning, yet they sometimes loop without stopping, their chains of thought are not a reliable record of what they compute, and we lack an account of how local computations are coordinated into reasoning trajectories. Here we procedurally generate SchemaBench, multi-phase schema tasks that vary roles, temporal structure, and complexity and allow ground-truth parsing of reasoning traces. We evaluate 10 LLMs' behavior and the residual-stream activations of Phi4-mini-R and Qwen3-4B. Their latent computation is organized at three levels: a) task-general operations conduct local computations with a shallow grammar, b) schema-level representations of roles, phases, and progress bias which operations follow, c) slower control states track the switching and completion of reasoning. Across levels, reasoning is organized in a low-dimensional subspace of the residual stream, where a state-dependent vector field predicts the next latent transition and reasoning 5-10 steps ahead. Causally injecting a STOP control-vector into Phi4-mini-R ends traces that could otherwise loop to the token cap, shortens response length by 4000+ tokens without sacrificing accuracy, and transfers to an unseen mathematical task. LLM reasoning is thus organized as multiscale predictive dynamics within a shared low-dimensional field, which makes hidden reasoning observable and controllable.

1 INTRODUCTION

Language models routinely perform long, multi-step reasoning tasks, yet we still lack an account of how local computations are selected and how they interact and evolve to form coherent reasoning trajectories to solve a problem. This limited understanding together with unexpected behaviors have led to a wave of recent studies questioning the reliability (Kwon et al., 2026), observability (Turpin et al., 2023; Chen et al., 2025c; Guan et al., 2025; Arcuschin et al., 2025), and safety of these models (Wang et al., 2025; Zhou et al., 2025; Korbak et al., 2025). Models sometimes spend compute on unproductive paths and get stuck in repetitive loops (Chen et al., 2025b; Sui et al., 2025; Huang et al., 2025; Duan et al., 2026; Xu et al., 2022), or generate reasoning trajectories that are not monitorable in their chain of thought (Baker et al., 2025; Pfau et al., 2024). This can lead to unsafe behaviors such as reward hacking, deception, and harmful autonomous actions in controlled evaluations (Greenblatt et al., 2024; Meinke et al., 2024; Lynch et al., 2025).

Limitations of steering approaches. While existing research offers a rich tool kit for steering model behavior through internal representations, these approaches largely operate at the level of global properties, which span steering persona and character traits (Zou et al., 2023; Turner et al., 2023; Chen et al., 2025a), truthfulness (Li et al., 2023), refusal (Arditi et al., 2024), function vectors (Todd et al., 2024), and, for reasoning models, the length and budget of the reasoning process (Muennighoff et al., 2025; Aggarwal & Welleck, 2025). Recent work succeeds in relatively finer scales of steering directions in the residual stream that elicit predefined behaviors, e.g., backtracking, uncertainty estimation, and example testing (Venhoff et al., 2025), or that scale overall reasoning effort and depth (Huang et al., 2025; Fang et al., 2026). However, they do not demonstrate observability and control over multiple scales of internal layer computation, from fine-grained computing operations to larger task-related parameters even slower control states.

Limitations of existing tasks and benchmarks. Most existing benchmarks and tasks do not provide ground truth to identify step-by-step reasoning to label the internal dynamics, and few interpretabil-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

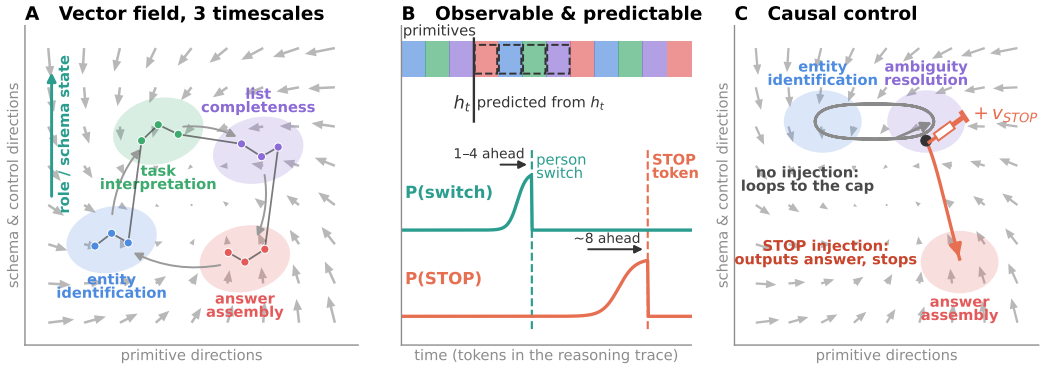


Figure 1: **Reasoning unfolds as multiscale predictive latent dynamics.** (**A**) A fitted vector field (grey arrows) predicts step-by-step changes in the residual stream h_t in a given layer (16) from time t to $t+1$. Three levels of representation are observed within the field. Reasoning operations unfold as a movement through clusters that each elicit a given computation (primitive operation) in the reasoning trace, 2-3 tokens each. Task-based states (e.g., a costumer role in a restaurant schema) occupy fixed directions in the field, while slower global control states determine when reasoning switches or stops. (**B**) Linear readouts of the latent state h_t predict future primitives (dashed outlines) and anticipate control state transitions (1-4 primitives before a person switch, 8 primitives before the STOP). (**C**) Injecting the direction of a control state, e.g. v_{STOP}, at layer 16 moves the trace to that state, e.g., injecting STOP forces the model to stop looping, assemble the answer, output it, and stop.

ity methods are designed to characterize the nonlinear, dynamic evolution of representations over time (Bricken et al., 2023; Templeton et al., 2024; Ameisen et al., 2025). While recent readout methods reach intermediate representations and unverbalized states (Gurnee et al., 2026; Fraser-Taliente et al., 2026), they do not characterize how those states evolve. Therefore, a) understanding how model reasoning evolves in their residual stream that generates reasoning and output tokens, and b) whether relevant representations are sufficiently low-rank to allow online monitoring of internal processes, and c) redirect and stop unwanted ones, are particularly important for reliability, efficiency, online monitoring, and control of safe behavior.

We address these aims with three contributions. First, we introduce SchemaBench, a new benchmark for schema-based reasoning tasks. Introduced nearly a century ago (Bartlett, 1932), schema tasks offer complex yet controlled tasks with multiple roles, interactions, and temporally extended order and phases to completion (Minsky, 1975b; Schank & Abelson, 1977; Rumelhart, 1980) and have been studied across the neurosciences (Tse et al., 2007; van Kesteren et al., 2012; Baldassano et al., 2018). We provide a detailed analysis of the schema inference abilities in 10 LLMs. Second, we analyze the latent representations of two of these LLMs while solving the SchemaBench task, and identify a lower dimensional shared field where representations and operations are coordinated at multiple temporal scales: local computational operations, schema-related states, and global control states that occur around transition between major reasoning events and before the model prepares to stop reasoning. Third, we causally extract the control state direction and inject it into the internal layers of an LLM (Phi4-mini-reasoning) to STOP the model's reasoning behavior. Moreover, we show that causally injecting the STOP state transfers to a very different mathematical task.

2 Related Work

Interpretability and Representational Analysis. Most interpretability approaches assume that meaningful model states and task variables are represented in the model latent representations, and probing directions or subspaces in activation space can approximate them (Park et al., 2024; Ameisen et al., 2025). Activation steering approaches aim to steer model behavior by causally intervening in the intermediate layers via linear steering (Todd et al., 2024; Hendel et al., 2023; Liu et al., 2024), manifold steering (Wurgaft et al., 2026). AlgoTrace identifies recurrent computational operations or algorithmic primitives and injects their associated top attention head vectors into a model to induce associated behavior (Eberle et al., 2025; Lippl et al., 2026). J-space and natural language

autoencoders explore non-verbalized representations in intermediate layers and show they can be causally induced or edited (Gurnee et al., 2026; Fraser-Taliente et al., 2026). What remains less known is the dynamics linking these objects over reasoning time, e.g., how persona level steering affects local computations, which representations predict the trajectory of computations that follow, and how slower control states reorganize the trajectory. Here we ask how different representations, operations, and task variables evolve and are coordinated over time, how far the latent sequence of reasoning is predictable, and whether we can steer global task-general states of a model.

Cognitive science and Neuroscience. A schema is a structured representation of a familiar situation, e.g., a Cafe or a restaurant, which often includes multiple interaction roles, entities, events, relations, and a sequence of events that unfold in order (Bartlett, 1932; Rumelhart et al., 1986). Scripts are the temporal sequence component of schema, specifying the events and role-specific actions that are expected to unfold over time in a familiar situation (Schank & Abelson, 1977). In recent decades, modeling, reinforcement learning (RL), and neuroimaging research has advanced the study of schema (Baldassano et al., 2018; Gilboa & Marlatte, 2017), offered accounts of binding arbitrary fillers to abstract roles and how this generalizes to unseen roles and fillers (Chen et al., 2021). Role-entity binding and event schemas have also been studied in language models (Feng & Steinhardt, 2024; Dror et al., 2023). However, we are not aware of public schema benchmarks with procedurally generated and variable number of roles, fillers, and many tasks to study complex multi-step reasoning in language models.

3 Methods and Results I: SchemaBench and behavioral results

SchemaBench: A Benchmark For Testing Schema-based reasoning. We introduce a new benchmark for testing schema-based reasoning in LLMs. SchemaBench presents short narrative passages adhering to one of two schemas (restaurant or airport), followed by questions about this passage that are designed to probe schema understanding. Crucially, these questions probe schema understanding in a holistic manner, by asking the models to 1) identify the schema and the different roles each person plays ("Identify" task), 2) identify violations of schema structure if any ("Spot error"), and 3) leverage their understanding of the schema to predict the next event or the next action by a particular person ("Predict next"). In designing the task generation pipeline, we were guided by three important principles: 1) detailed and reliable ground truth annotations, 2) careful controls against reasoning shortcuts, and 3) easy annotation of the generated reasoning traces.

Our pipeline combines symbolic and LLM-driven generation and verification, which allows for naturalistic stories (which symbolic programs, by themselves, cannot produce) with reliable ground truth and controls for potential confounders (which LLMs, by themselves, cannot guarantee). To control for reasoning shortcuts, our recipe specified whether persons and objects were explicitly named (e.g., waiters and menus are heavily associated with restaurants whereas "Anne" and "a card listing the options" are non-revealing). Additionally, we generated task variants that scramble the narrative order to separate the chronological order of events (which this perturbation does not effect) from the order of presentation. We further vary the overall detail mentioned in the stories, whether additional unrelated events are included, and whether multiple people serve in the same role, or whether one person serves in multiple roles (see Appendix B for detail).

Crucially, we generate a detailed annotation of the dataset, allowing us to identify at what point during the story particular roles become identifiable. This allows us to leverage decoding analyses of roles during prompt reading. Similarly, since each role is associated with a clear, symbolically-parsable name or description, we can reliably determine which part of the schema the model is reasoning about in its reasoning trace. Taken together, SchemaBench enables reliable and detailed investigations of models' schema understanding and provides a range of tools that allow for detailed and robust analysis of models' latent representations and reasoning traces.

Behavioral Evaluation on SchemaBench. We evaluated ten models across four model families: Phi (Phi-4-mini-reasoning, Phi-4, Phi-4-reasoning, and Phi-4-reasoning+), Qwen (Qwen3-4B, Qwen3-14B, and Qwen3.8-27B, DeepSeek-R1-Distill-Qwen-14B), DeepSeek (DeepSeek-V4-Pro), and GPT-5.5 on SchemaBench. We found that all models were able to solve SchemaBench with some accuracy (especially taking into account that "chance" accuracy here is virtually 0% for each task given the many possible responses) (Fig. 2A). GPT-5.5 generally had the highest performance. Among the smaller open-weight models, Phi-4-reasoning+ and Qwen3.8-27B performed the best,

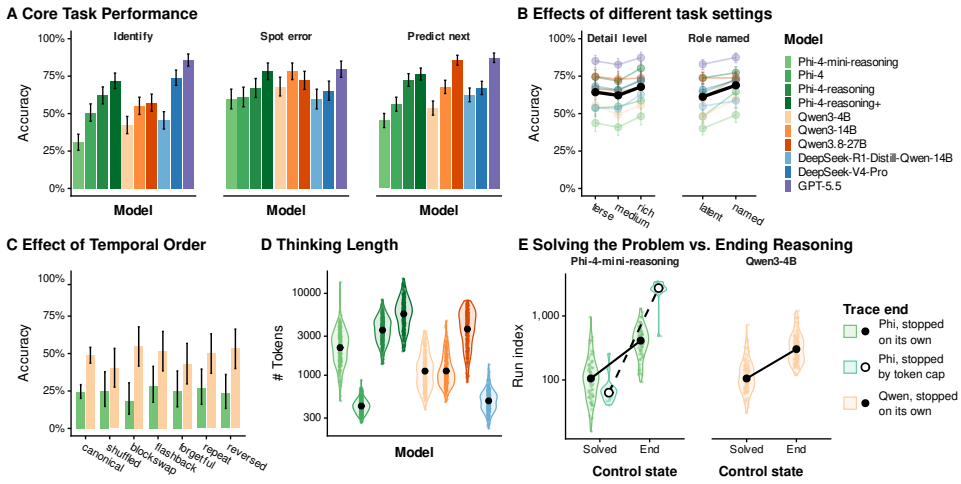


Figure 2: Behavioral evaluation on SchemaBench. (**A**) Reasoning accuracy across different tasks and models. (**B**) Accuracy across selected task complexity settings. (**C**) Accuracy of selected models (Phi-4-mini-reasoning and Qwen3-4B) on passages where the narrative order is systematically varied. (**D**) Response length (log scale) across different models. Note that we do not have access to the thinking traces for GPT-5.5 and DeepSeek-V4-Pro. We depict the geometric mean. (**E**) Comparison of when the model already solves the problem versus when they actually end its reasoning. Run index depicts the number of meta-cluster switches. Note that Phi-4-Mini-Reasoning models sometimes do not terminate their reasoning within the token cap (8192 tokens; highlighted with white dots). Run index is plotted on a log scale and we depict the geometric mean.

with Phi-4-reasoning+ performing better at identifying schemas, Qwen3.8-27B performing better at predicting the next event or action, and both performing comparably at spotting schema errors. For each model family, there was a clear performance gradient: larger models outperformed smaller models and more finetuned models outperformed less finetuned models. We found that performance depended on the different task properties (Fig. 2B; see also Fig. 7C). In particular, models performed better when the roles of different people were explicitly named. Perhaps surprisingly, models performed slightly better for richly detailed stories. We also found that model performance was not strongly affected by the order with which events were presented (Fig. 2C). Moreover, one person taking on multiple roles also resulted in a drop in performance, while several people taking on the same role sometimes led to a performance improvement (Fig. 10).

Next, we investigated whether reasoning efficiency differed between different models. We found that larger models generally used more tokens. Moreover, the Phi models used substantially more tokens than the Qwen models (Phi-4-reasoning+ used a median 5782 tokens while Qwen3.8-27B used a median 3980 tokens ($p < 10^{-10}$); similarly, Phi-4-mini-reasoning used a median 2285 tokens while Qwen3-4B used a median 1118 tokens ($p < 10^{-10}$); Fig. 2D). Finally, we determined the location in the reasoning trace at which the models arrived at the correct solution for the "Identify" task, by using an LLM judge to determine whether the model had correctly listed both the schema and all role-person pairings. We found that both Phi-4-reasoning-mini and Qwen3-4B arrived at the correct solution long before they stopped reasoning: both Phi and Qwen reach the correct solution at a median of 106 runs (runs are defined as the number of meta-cluster switches) but then continue reasoning for a median of 265 runs (for Phi) and 176 runs (for Qwen) (Fig. 2E). In some cases (n=17), Phi did not finish reasoning by the time it reached the maximal token amount, despite having solved the task much early (median of 63 runs). This is consistent with prior reports of overthinking in large language models (Sun et al., 2026; Zhou et al., 2026).

4 Methods and Results II: 3 Scales of Latent Representation

To analyze the latent computations in LLMs solving schema-based reasoning tasks, we focused on three different levels of representation and computation. 1) We extracted "algorithmic primitives", minimal latent computational operations that elicit the model's reasoning process. 2) We analyzed

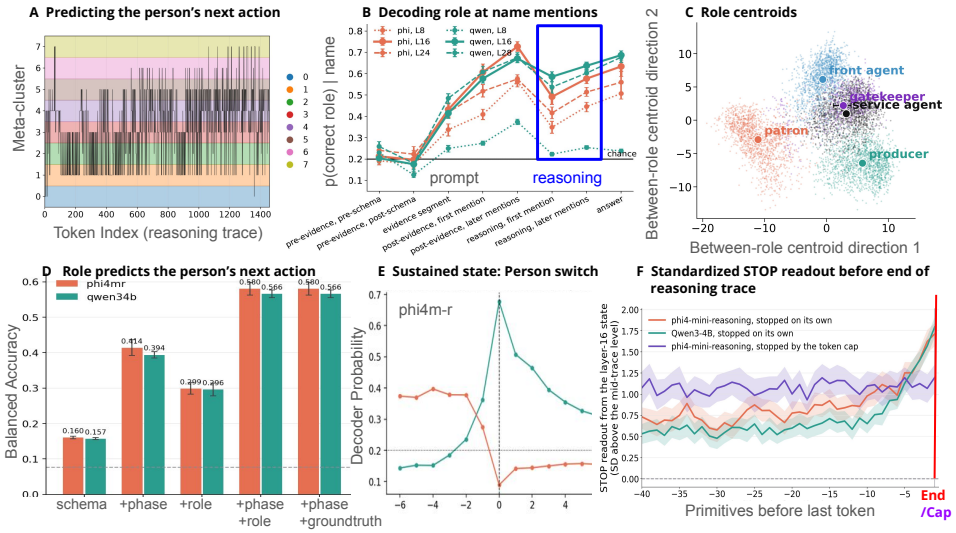


Figure 3: **Schema-based and control states in the latent. (A)** Primitive label at every token of one reasoning trace. **(B)** A decoder was trained only on role words and tested at a person's name mentions. The person's correct role is at chance before it is mentioned in the prompt, and rises to 0.73 / 0.67 after it is mentioned or inferred (Phi / Qwen, L16). **(C)** Role centroids of Phi's latent at name mentions. **(D)** Balanced accuracy for predicting a person's next action. Role adds to the schema and phase, and the role decoded from the latent gives nearly the same accuracy as ground truth. **(E)** Role decoder around a person switch (Phi). The incoming role can be decoded one to two primitives before the name is mentioned, and decoding the outgoing role drops below chance. **(F)** Held-out STOP readout aligned at the last token of the trace. Traces that stop rise over the last eight primitives (+0.60 / +1.03 SD); Phi traces stopped by the token cap sit one SD above their mid-trace level throughout and never rise.

how the model encoded different schema states and its predictions of future states. 3) We identified higher-level control states that track switches between schema states, preparation to stop, and outputting the final response to terminate the reasoning process.

Local Dynamics of Reasoning: Computational Operation Primitives. To analyze the latent computational operation primitives in LLMs solving schema-based reasoning tasks, we applied methods introduced in AlgoTrace (Lippl et al., 2026). Specifically, we extract algorithmic primitives (i.e., latent operations observed in a reasoning process) through clustering the LLMs' latent representations during reasoning, and then annotate these primitives using an automated LLM pipeline (see Appendix C). We found that distinct primitives correspond to discrete, interpretable steps relevant for identifying schema-relevant information, such as "Entity action mapping" and "Scenario norm comparison." After extracting the latent algorithmic primitives, we next examine how schema information and primitives are processed and composed throughout reasoning.

Slower Schema-Level State. First, we test to what extent schema states can be decoded from h_t, fitted inside each training fold and under a held-out item condition to control against memorizing passages. For Phi/Qwen at layer 16, we find the following decoding accuracies: role (e.g., chef) at 0.929 / 0.944 (chance 0.200) over all mentions, prop (e.g., menu) at 0.997 / 1.000 (0.125), event at 0.412 / 0.416 (0.077), and schema (restaurant, airport) at 0.880 / 0.894 (0.500). Additional settings are summarized in Tab. 5 in Appendix G. Schema-level variables are decoded at ceiling given the first 48 PCs of the residual state (total dimensions: 2056 for qwen3-4b, 3072 for phi4m-r).

Next, we asked when binding between role and a person's name occurred in the model computations. We trained a decoder only on role-word tokens and tested at name mentions to test how well it can assign the correct role, depending on position along different steps in the prompt, reasoning, and answer segments (see Fig. 3B). Before the evidence segment, layer 16 role-at-person decoding is at chance (0.20) in both Phi and Qwen. It rises to 0.43 / 0.42 after the evidence is introduced and reaches 0.73 / 0.67 after re-mentions by the end of the prompt segment. We observe a drop in decoding probability at reasoning onset, with the model largely recovering peak performance until the final answer. Thus, roles become decodable from a person's name only after role evidence is

Table 1: Objects and definitions used in the latent analyses.

Term	Definition
Latent state, residual stream	$h_t \in \mathbb{R}^d$, typically layer-16 residual at the last token of primitive occurrence t, read through its first 48 principal components. Its change is $\Delta h_t = h_{t+1} - h_t$. Some analyses include layers 8 and 24.
Primitive	A minimal latent reasoning operation extracted from layer-16 using AlgoTrace. There are 35 total primitive clusters. Their 8 meta-clusters per model, with centroids μ_z, are the alphabet $z \in \{1, \ldots, 8\}$.
Schema-based state	The active person's role, the schema phase, and progress, read from h_t by linear decoders and held by the field as slow directions.
Control state	Slower global state, e.g., nearing a switch of the active person, flagging that all role-name pairs are solved, or approaching STOP. Readable from h_t 1-8 primitives (3-15 tokens) before appearing in the reasoning trace.
Vector field	$F(h) = \mathbb{E}[\Delta h_t \mid h_t = h]$, a held-out regression from h_t to Δh_t, split into a relaxation toward the mean, $a + s \odot h_t$, and remainder $G(h_t) = F(h_t) - (a + s \odot h_t)$.
Successor vector field	$S_\gamma(h_t) = \mathbb{E}\Big[\sum_{k \geq 1} \gamma^{k-1}(h_{t+k} - h_t) \,\Big|\, h_t\Big]$, the discounted future displacement, with the k-step field $\mathbb{E}[h_{t+k} - h_t \mid h_t]$ as its components.
Control vector	Extracted control state vector that can be injected to another task. Here, v_{STOP}, the residual difference between the moments before a spontaneous STOP and earlier moments in the same response, injected as $h_{16} \leftarrow h_{16} + \alpha\, v_{\text{STOP}}$, $\alpha = 0.25$.

introduced and is reinstated at later mentions of the name. This is consistent with a slower schema-level state that persists across multiple primitive operations and carries role information forward to bias subsequent operations.

We further test how well the person's next action can be decoded from the schema, computing mutual information see Fig. 3D. With chance level at 0.08, schema alone reaches 0.16 / 0.16 balanced accuracy in Phi/Qwen. We then ask to what extent additional schema-level variables, specifically phase and role, improve prediction. Under item-grouped cross-validation, adding the global canonical phase g_t of the episode at the person's current action raises performance to 0.41 / 0.39, while adding the decoded role posterior $\hat{r}$ at the first name mention reaches 0.30 / 0.30. Combining both yields 0.58 / 0.57, matching the performance of the gold role ceiling. As a null, permuting the decoded role across persons within schema yields 0.43 / 0.39. To quantify how much sequential structure extracted primitive sequences exhibit, we measure the conditional entropy (bits per next primitive) of predictors of order n (see Fig. 12 in the Appendix), where n is the number of preceding primitives used for prediction: lower scores indicate that the recent history makes the next primitive more predictable, thus demonstrating local predictability. Across the two settings, bits per next primitive decrease from 3.20 / 2.97 at order 0 to 2.03 / 1.95 at order 1 and 1.61 / 1.62 at order 2, with little further reduction at order 3 (1.58 / 1.57). Moreover, lag-k mutual information remains higher than that of an order-1 Markov surrogate out to lag 20, indicating dependencies that cannot be explained by first-order transitions alone.

Longer-Horizon Task-General Control States. To distinguish functional global control states from other schema states and primitives, we define control states using complementary top-down and bottom-up analyses. Top-down, from labels to the latent, we define candidate control variables on the labeled trace: the upcoming switch of the active person, the solved flag, the approach to STOP, and the current schema phase. We then ask whether these variables are represented in the layer-16 representations. Linear readouts recover the upcoming switch within 1 to 4 primitives with AUCs of 0.89 / 0.90 and 0.81 / 0.82. Bottom-up, from the field to labels, we perturb 6,000 held-out mid-trace (10%–90%) states along candidate directions while keeping the primitive fixed, and ask how the learned routers and successor field predict the future. Event directions produce specific shifts toward their corresponding future profiles, while random and schema-identity directions produce little or non-specific change. We validate these predictions against naturally occurring states matched on primitive, phase, and trace position. The field thus identifies these directions as dynamically privileged, independently of their semantic labels. Section 6 causally tests control states and their transfer to an unseen task.

5 METHODS AND RESULTS III: VECTOR FIELD, SUCCESSOR VECTOR FIELD

The first 48 principal components carry about half of the latent's variance (46% Phi, 56% Qwen), but they carry most of the predictive structure. The vector field fitted within the 48 PCs explains 100% of the variance of the full-residual field for Phi (0.50 for 48, 0.50 for all 3072 dimensions) and 95% for Qwen (0.52 for 48D, 0.55 for all 2560). Moreover, the field's primitive-routing direction at the eight metacluster centroids captures 95% of the full-residual's prediction for Phi (0.63 for 48D, 0.66 for all) and 100% for Qwen (0.66 / 0.66), while the first 4 dimensions already capture 0.61 for Phi and 0.63 for Qwen.

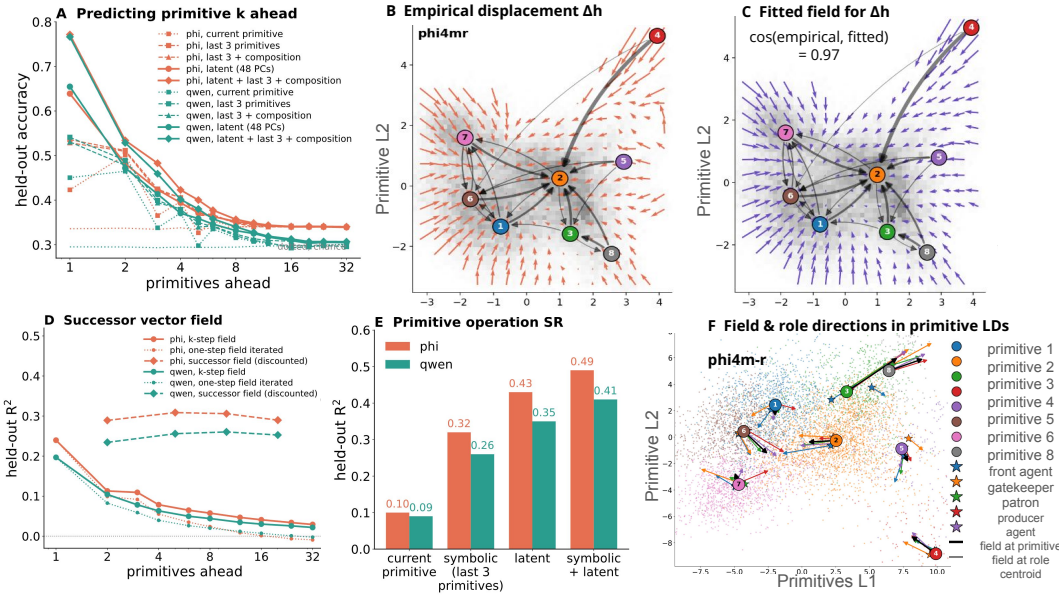


Figure 4: **Predicting future reasoning.** (**A**) Predicting which primitive appears k primitives later using symbolic history (last three primitive) and latent at the end token of current primitive, layer-16's first 48 PCs. Dotted lines mark chance, the most frequent primitive, 0.34 for Phi and 0.30 for Qwen. (**B**) Empirically measured field, vectors of movements at every point in the latent space projected two primitive LDs. (**C**) Fitted field for Δh (**D**) Successor vector field. Circles show how well the k-step vector field, that is fitted from the current latent, predicts where the latent will be k primitives later (held-out R^2, with the return to the mean removed). Diamonds show prediction accuracy for the successor vector field, which predicts the discounted sum of the coming changes, so that nearer primitives count more than distant ones, at $\gamma = 0.8$ and 0.95, reaching about 5 and 20 primitives ahead. (**E**) Successor representation over primitives. Each bar is the held-out R² for the discounted future primitive occupancy at $\gamma = 0.9$ over the next 32 primitives, from the current primitive alone, the last-3-primitives symbolic history, the current latent (layer 16, 48 PCs), and symbolic history plus the latent. **F**) Field and role directions plotted in two primitive LDs.

A Predictive Field over Multi-scale Internal Computations. In the token-by-token dynamics of the Transformer's hidden layer, consider a given input sequence $x_{1:T}$. For a subsequence $x_{1:t}$, $t \leq T$, the hidden representations in a particular layer of the Transformer, $H(x_{1:t}) \in \mathbb{R}^{t \times d}$,where d is the residual dimension. Thus, the current token's representation is: $h_t := H_t(x_{1:t}) \in \mathbb{R}^d$.

As reasoning unfolds, the change in this latent representation, Δh, given transition from the current to the next state, may depend on both the current representation and the preceding context. There are two hypotheses about this change: a) it can be explained in terms of a vector added to all points $h_{t+1} = h_t + b$), or b) it can be explained with a vector field, where different points may move in different directions depending on the structure of the field, $h_{t+1} = h_t + F(h_t)$, with $F(h) = Ah + b$ in the affine fit we use. Here $F : \mathbb{R}^d \to \mathbb{R}^d$ defines a *vector field*, which assigns a direction of motion to every point in the representation space. The change captured by the field can be decomposed into a state-dependent component and a context-dependent residual: $h_{t+1} = h_t + F(h_t) + R(x_{1:t})$. The residual $R(x_{1:t})$ captures context-dependent changes that cannot be explained by the current state alone. In what follows we show that the field explains Δh better than vector addition (Fig. 4B,C). This suggests that the sequence of latent operations traces a trajectory through latent representation

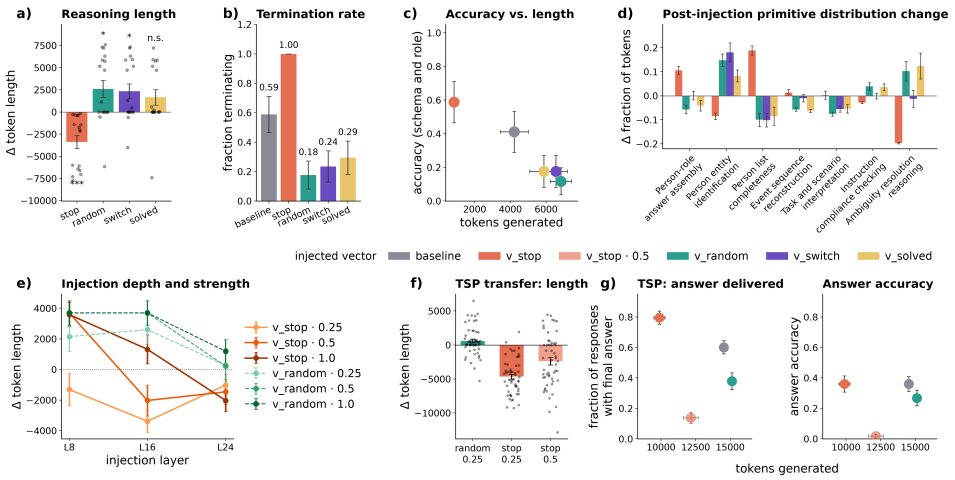


Figure 5: **Causal intervention of stop control vector. (a–e)** the stop vector v_{STOP} is injected into the layer-16 residual stream of Phi-4-mini-reasoning model solving the identify task. **(a)** change in response length. **(b)** proportion of responses terminating. **(c)** accuracy against tokens generated. **(d)** post-injection change in distribution of primitives. **(e)** sensitivity of injection across injection depths and strengths. **(f–g)** Transfer the stop vector to traveling-salesman problems (TSP). **(f)** change in response length. **(g)** answer delivery and accuracy. v_{RANDOM} is the random vector control, v_{SWITCH} is the switch vector control, and v_{SOLVED} is the solved vector control.

space. Moreover, the dynamics of a shared field $F(h_t)$ can predict latent representations at multiple scales noted in C.1. We find that the field predicts both slow state-dependent changes and local direction of motion at the 2-token computational operation primitive scale.

Successor Vector Field Predicts Future Latents and Reasoning. A successor representation (SR) is the discounted occupancy of the states that follow a given state (Dayan, 1993), and multi-scale versions of it have been found along the hippocampal and prefrontal hierarchy (Brunec & Momennejad, 2022). We compute successor representation over the vector field, which evaluates how well the field predicts the discounted future of the latent rather than a static representation The successor vector field $S_\gamma(h_t)$ is fitted to the discounted sum of the latent's coming changes (Table 1). It predicts the sequential part of the latent at held-out R^2 of 0.31 / 0.26 five primitives ahead and 0.29 / 0.25 twenty ahead, whereas iterating the one-step field reaches only four primitives (Fig. 4D). As a successor representation over primitives, the latent predicts the next 32 primitives at R^2 0.43 / 0.35 against 0.32 / 0.26 for the last three symbolic primitives (Fig. 4E).

Successor Vector Fields Show the Field is Multi-scale and Multi-step. Inspired by fMRI analyses of multi-scale successor representations (Brunec & Momennejad, 2022), we asked whether the field predicts at more than one horizon. The eight primitive clusters are laid out by their futures five to ten primitives ahead (0.79 / 0.81), not by their next step (0.05 / 0.18). The current primitive sets the near and the schema-based state the far future: the field follows the primitive's predicted future more at five primitives ahead (0.14 / 0.24 vs 0.10 / 0.16) and the two equally at twenty (0.11 / 0.17 each).

6 Methods and Results IV: Causal injection of STOP control

So far, we have found that the latent states identified above correlate with different aspects of model reasoning. To evaluate whether they are *causally relevant*, we performed intervention experiments by constructing a v_{STOP} vector and injecting it into Phi-4-Mini-Reasoning. We construct v_{STOP} by contrasting the residual activity immediately before a spontaneous stop in a reasoning trace with residual activity earlier in the same response, and then extracting the residual stream projection corresponding to the top 2% most influential attention heads (as in prior work; Todd et al., 2024; Lippl et al., 2026). We identified reasoning traces where the model falls into long recurring loops, indicating a failure to stop spontaneously, and injected v_{STOP} to see if it could induce stopping. We compared against three controls: v_{RANDOM} (random vectors of the same dimension), v_{SWITCH}

(constructed using states where the model switches from reasoning about one person to another) and v_{SOLVED} (constructed using states where the task is solved) (see Appendix D).

Stop vector induces shorter reasoning without sacrificing accuracy. Injecting the stop vector produces significantly shorter responses (837 vs. 4218 tokens, p=1.5e-5) and causes Phi to finish reasoning in 100% of responses, compared to a baseline termination rate of 59% (Fig. 5A,B). Notably, injecting stop vectors did not compromise accuracy (0.59 with injection vs. 0.41 without, p=0.41, Fig. 5C). In contrast, three other control injections lengthened responses and reduced accuracy, highlighting the specificity of the identified vector. We further analyzed the distribution of reasoning primitives after injection and found that injecting v_{STOP} induces more tokens on *answer assembly* and *person list completeness*, with fewer tokens to *person entity identification* and *ambiguity resolution*, indicating that the stop vector may induce more efficient reasoning primitives (Fig. 5D). Finally, we found that the stop vector's effect is robust only within a specific parameter range, with injection in middle layers and smaller injection strength being the most effective (Fig. 5E). This suggests that v_{STOP} affects reasoning via a nonlinear downstream effect on model computation.

Stop vector transfers to very different task. To test whether stop vectors transfer to other tasks, we performed injection on the traveling salesman problem (TSP) (Lippl et al., 2026). Injecting the stop vector when solving TSP caused the responses to be 31% shorter (-4908 tokens, p=3.2e-10, Fig. 5F), whereas the effect of random vector was not significant. The responses with stop vectors produce final answers more often (0.796 vs. 0.602, p=0.006), with accuracy not significantly different from the baseline (p=0.99, Fig. 5G), which may be because TSP tasks require generating and evaluating more candidate solutions. Overall, we demonstrated that the stop vector can effectively reduce response length and improve reasoning efficiency, and its effect can transfer to other tasks.

7 Discussion

We design schema-based reasoning tasks to evaluate multi-step compositional reasoning in language models, and show that it is coordinated by multiscale computations, with local primitive operations that change at a faster scale, every 2-3 tokens, and slower task-related and control states at 3-5 primitives (6-15 tokens). Notably, these multiscale representations are decodable from a shared low-dimensional field in the residual stream (48 out of 2500+ dimensions). The field can predict future changes in the residual stream up to 16 operations ahead. Control state representations can be causally injected to elicit a control state that expresses multi-step reasoning primitive computations, and STOPs model reasoning, eliciting a response. This causal intervention transfers to a very different task family and decreased reasoning traces by 4000 tokens without harming performance.

AI Safety Implications. Recent work raises concerns about observability and monitorability of model reasoning, warning about loss of control and inability to stop them (Stix et al., 2025; Bengio et al., 2025). Our results offer proof of concept that latent computations can be both monitored and controlled from a low-dimensional field (see also Gurnee et al., 2026). This in principle suggests an online monitor can be built to both monitor the model's current reasoning computations, and also to predict what the model may "think" many steps ahead. Our causal injection results show that the same latent states can steer the model, e.g., injecting the stop control vector at layer 16 cuts response length without lowering accuracy, and transfers to a different task family (mathematical reasoning).

Future Directions: Monitoring, Predicting, and Controlling Hidden Computations. Our results suggest a small number of slow computational states coordinate faster local operations in model reasoning. For safety these observable and controllable global states of reasoning models offer a complement to output monitoring and post-training. Moreover, it offers a direct handle on the non-termination pathology displayed by models in the Phi family (Fig. 5), saving thousands of tokens in the reasoning process. Future directions include testing state-space models as well as loop transformers for state-dependent field dynamics.

Limitations. We fit a vector field to approximate the complex multi-layer computations of the Transformer, but do not claim that this constitutes an autonomous dynamical system (see Appendix E). Extending our work to looped Transformers and other recurrent models (Giannou et al., 2023) will allow us to fully benefit from dynamical systems tools like saddle point and attractor analysis. We identified putative algorithmic primitives through an approach that was independent from the vector field. Future work could use the vector field directly to inform the identified algorithmic primitives.

8 AI USE STATEMENT.

Claude Code and ChatGPT, various versions, were used throughout the project to assist in coding the analysis, improving figures, equations, and tables, and give feedback on writing. We have reviewed all AI-assisted work and take responsibility for the final content of this work.

REFERENCES

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, James R. Lee, Yin Tat Lee, Yuanzhi Li, Weishung Liu, Caio C. T. Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Xin Wang, Rachel Ward, Yue Wu, Dingli Yu, Cyril Zhang, and Yi Zhang. Phi-4 technical report, 2024. URL https://arxiv.org/abs/2412.08905.

Marah Abdin, Sahaj Agarwal, Ahmed Awadallah, Vidhisha Balachandran, Harkirat Behl, Lingjiao Chen, Gustavo de Rosa, Suriya Gunasekar, Mojan Javaheripi, Neel Joshi, Piero Kauffmann, Yash Lara, Caio César Teodoro Mendes, Arindam Mitra, Besmira Nushi, Dimitris Papailiopoulos, Olli Saarikivi, Shital Shah, Vaishnavi Shrivastava, Vibhav Vineet, Yue Wu, Safoora Yousefi, and Guoqing Zheng. Phi-4-reasoning technical report, 2025. URL https://arxiv.org/abs/2504.21318.

Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning. *arXiv preprint arXiv:2503.04697*, 2025.

Emmanuel Ameisen, Jack Lindsey, Adam Pearce, Wes Gurnee, Nicholas L. Turner, Brian Chen, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, et al. Circuit tracing: Revealing computational graphs in language models. *Transformer Circuits Thread*, 2025. https://transformer-circuits.pub/2025/attribution-graphs/methods.html.

Iván Arcuschin, Jett Janiak, Robert Krzyzanowski, Senthooran Rajamanoharan, Neel Nanda, and Arthur Conmy. Chain-of-thought reasoning in the wild is not always faithful. *arXiv preprint arXiv:2503.08679*, 2025.

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. In *Advances in Neural Information Processing Systems*, volume 37, 2024. arXiv:2406.11717.

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.

Christopher Baldassano, Uri Hasson, and Kenneth A. Norman. Representation of real-world event schemas during narrative perception. *The Journal of Neuroscience*, 38(45):9689–9699, 2018. doi: 10.1523/JNEUROSCI.0251-18.2018.

Frederic C. Bartlett. *Remembering: A Study in Experimental and Social Psychology*. Cambridge University Press, 1932.

Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael K Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xLSTM: Extended long short-term memory. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=ARAxPPIAhq.

Yoshua Bengio, Sören Mindermann, Daniel Privitera, Tamay Besiroglu, Rishi Bommasani, Stephen Casper, Yejin Choi, Philip Fox, Ben Garfinkel, Danielle Goldfarb, Hoda Heidari, Anson Ho, Sayash Kapoor, Leila Khalatbari, Shayne Longpre, Sam Manning, Vasilios Mavroudis, Mantas Mazeika, Julian Michael, Jessica Newman, Kwan Yee Ng, Chinasa T. Okolo, Deborah Raji, Girish Sastry, Elizabeth Seger, Theodora Skeadas, Tobin South, Emma Strubell, Florian Tramèr, Lucia Velasco, Nicole Wheeler, Daron Acemoglu, Olubayo Adekanmbi, David Dalrymple, Thomas G. Dietterich, Edward W. Felten, Pascale Fung, Pierre-Olivier Gourinchas, Fredrik

Heintz, Geoffrey Hinton, Nick Jennings, Andreas Krause, Susan Leavy, Percy Liang, Teresa Ludermir, Vidushi Marda, Helen Margetts, John McDermid, Jane Munga, Arvind Narayanan, Alondra Nelson, Clara Neppel, Alice Oh, Gopal Ramchurn, Stuart Russell, Marietje Schaake, Bernhard Schölkopf, Dawn Song, Alvaro Soto, Lee Tiedrich, Gaël Varoquaux, Andrew Yao, Ya-Qin Zhang, Fahad Albalawi, Marwan Alserkal, Olubunmi Ajala, Guillaume Avrin, Christian Busch, André Carlos Ponce de Leon Ferreira de Carvalho, Bronwyn Fox, Amandeep Singh Gill, Ahmet Halit Hatip, Juha Heikkilä, Gill Jolly, Ziv Katzir, Hiroaki Kitano, Antonio Krüger, Chris Johnson, Saif M. Khan, Kyoung Mu Lee, Dominic Vincent Ligot, Oleksii Molchanovskyi, Andrea Monti, Nusu Mwamanzi, Mona Nemer, Nuria Oliver, José Ramón López Portillo, Balaraman Ravindran, Raquel Pezoa Rivera, Hammam Riza, Crystal Rugege, Ciarán Seoighe, Jerry Sheehan, Haroon Sheikh, Denise Wong, and Yi Zeng. International ai safety report, 2025. URL https://arxiv.org/abs/2501.17805.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, et al. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features.

Iva K. Brunec and Ida Momennejad. Predictive representations in hippocampal and prefrontal hierarchies. *Journal of Neuroscience*, 42(2):299–312, 2022.

Catherine Chen, Qihong Lu, Andre Beukers, Christopher Baldassano, and Kenneth A Norman. Learning to perform role-filler binding with schematic knowledge. *PeerJ*, 9:e11046, 2021.

Runjin Chen, Andy Arditi, Henry Sleight, Owain Evans, and Jack Lindsey. Persona vectors: Monitoring and controlling character traits in language models. *arXiv preprint arXiv:2507.21509*, 2025a.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Do NOT think that much for 2+3=? on the overthinking of long reasoning models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 9487–9499. PMLR, 2025b.

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*, 2025c.

Alejandro Cuadron, Dacheng Li, Wenjie Ma, Xingyao Wang, Yichuan Wang, Siyuan Zhuang, Shu Liu, Luis Gaspar Schroeder, Tian Xia, Huanzhi Mao, et al. The danger of overthinking: Examining the reasoning-action dilemma in agentic tasks. *arXiv preprint arXiv:2502.08235*, 2025.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624, 1993.

Christian Schroeder de Witt, Klaudia Krawiecka, Igor Krawczuk, Ben Hagag, William L Anderson, Peter Belcak, Ben Bucknall, Xiaohong Cai, Ayush Chopra, Doron Cohen, et al. Open challenges in multi-agent security: Towards secure systems of interacting ai agents. *arXiv preprint arXiv:2505.02077*, 2025.

Rotem Dror, Haoyu Wang, and Dan Roth. Zero-shot on-the-fly event schema induction. In Andreas Vlachos and Isabelle Augenstein (eds.), *Findings of the Association for Computational Linguistics: EACL 2023*, pp. 705–725, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-eacl.53. URL https://aclanthology.org/2023.findings-eacl.53/.

Zenghao Duan, Liang Pang, Zihao Wei, Wenbin Duan, Yuxin Tian, Shicheng Xu, Jingcheng Deng, Zhiyi Yin, and Xueqi Cheng. Circular reasoning: Understanding self-reinforcing loops in large reasoning models, 2026.

Oliver Eberle, Thomas Austin McGee, Hamza Giaffar, Taylor Whittington Webb, and Ida Momennejad. Position: We need an algorithmic understanding of generative AI. In *Forty-second International Conference on Machine Learning Position Paper Track*, 2025. URL https://openreview.net/forum?id=eax2ixyeQL.

Yi Fang, Wenjie Wang, Mingfeng Xue, Boyi Deng, Fengli Xu, Dayiheng Liu, and Fuli Feng. Controllable llm reasoning via sparse autoencoder-based steering. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 21290–21305, 2026.

Jiahai Feng and Jacob Steinhardt. How do language models bind entities in context? In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=zb3b6oKO77.

Kit Fraser-Taliente, Subhash Kantamneni, Euan Ong, Dan Mossing, Christina Lu, Paul C. Bogdan, Emmanuel Ameisen, James Chen, Dzmitry Kishylau, Adam Pearce, Julius Tarng, Alex Wu, Jeff Wu, Yang Zhang, Daniel M. Ziegler, Evan Hubinger, Joshua Batson, Jack Lindsey, Samuel Zimmerman, and Samuel Marks. Natural language autoencoders produce unsupervised explanations of llm activations. *Transformer Circuits Thread*, 2026. URL https://transformer-circuits.pub/2026/nla/index.html.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 41340–41391. Curran Associates, Inc., 2025. doi: 10.52202/085713-1380. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/3b01972cf31e6fa0fe29e4b8b5c2a0a1-Paper-Conference.pdf.

Angeliki Giannou, Shashank Rajput, Jy-yong Sohn, Kangwook Lee, Jason D. Lee, and Dimitris Papailiopoulos. Looped transformers as programmable computers. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Asaf Gilboa and Hannah Marlatte. Neurobiology of schemas and schema-mediated memory. *Trends in Cognitive Sciences*, 21(8):618–631, 2017.

Ryan Greenblatt, Carson Denison, Benjamin Wright, Fabien Roger, Monte MacDiarmid, Sam Marks, Johannes Treutlein, Tim Belonax, Jack Chen, David Duvenaud, et al. Alignment faking in large language models. *arXiv preprint arXiv:2412.14093*, 2024.

Xiangming Gu, Xiaosen Zheng, Tianyu Pang, Chao Du, Qian Liu, Ye Wang, Jing Jiang, and Min Lin. Agent smith: A single image can jailbreak one million multimodal llm agents exponentially fast. In *International Conference on Machine Learning*, 2024. arXiv:2402.08567.

Melody Y. Guan, Miles Wang, Micah Carroll, Zehao Dou, Annie Y. Wei, Marcus Williams, Benjamin Arnav, Joost Huizinga, Ian Kivlichan, Mia Glaese, Jakub Pachocki, and Bowen Baker. Monitoring monitorability, 2025.

Wes Gurnee, Nicholas Sofroniew, Adam Pearce, Mateusz Piotrowski, Isaac Kauvar, Runjin Chen, Anna Soligo, Paul Bogdan, Euan Ong, Rowan Wang, Ben Thompson, David Abrahams, Subhash Kantamneni, Emmanuel Ameisen, Joshua Batson, and Jack Lindsey. Verbalizable representations form a global workspace in language models. *Transformer Circuits Thread*, 2026. URL https://transformer-circuits.pub/2026/workspace/index.html.

Lewis Hammond, Alan Chan, Jesse Clifton, Jason Hoelscher-Obermaier, Akbir Khan, Euan McLean, Chandler Smith, Wolfram Barfuss, Jakob Foerster, Tomáš Gavenčiak, et al. Multi-agent risks from advanced ai. *arXiv preprint arXiv:2502.14143*, 2025.

Roee Hendel, Mor Geva, and Amir Globerson. In-context learning creates task vectors. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 9318–9333, 2023.

Yao Huang, Huanran Chen, Shouwei Ruan, Yichi Zhang, Xingxing Wei, and Yinpeng Dong. Mitigating overthinking in large reasoning models via manifold steering. In *Advances in Neural Information Processing Systems*, volume 38, pp. 113645–113670, 2025. doi: 10.52202/085713-3430.

Tomek Korbak, Mikita Balesni, Elizabeth Barnes, Yoshua Bengio, Joe Benton, Joseph Bloom, Mark Chen, Alan Cooney, Allan Dafoe, Anca Dragan, et al. Chain of thought monitorability: A new and fragile opportunity for ai safety. *arXiv preprint arXiv:2507.11473*, 2025.

Yongchan Kwon, Shang Zhu, Federico Bianchi, Kaitlyn Zhou, and James Zou. ReasonIF: Large reasoning models fail to follow instructions during reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 29149–29164. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.1456.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In *Advances in Neural Information Processing Systems*, volume 36, 2023. arXiv:2306.03341.

Samuel Lippl, Thomas Austin McGee, Kimberly Lopez, Ziwen Pan, Pierce Zhang, Salma Ziadi, Oliver Eberle, and Ida Momennejad. Algotrace: Algorithmic primitives and compositional geometry of reasoning in language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=ppirVueEj6.

Sheng Liu, Haotian Ye, Lei Xing, and James Zou. In-context vectors: making in context learning more effective and controllable through latent space steering. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Yaoyang Luo, Zhi Zheng, Ziwei Zhao, Tong Xu, Jielun Zhao, Wenjun Xue, Yong Chen, and Enhong Chen. Defending llm-based multi-agent systems against cooperative attacks with sentence-level rectification, 2026.

Aengus Lynch, Benjamin Wright, Caleb Larson, Stuart J. Ritchie, Sören Mindermann, Ethan Perez, Kevin K. Troy, and Evan Hubinger. Agentic misalignment: How llms could be insider threats, 2025.

Oliver Makins, Orazio Angelini, Zohreh Shams, and Mary Phuong. Multi-agent ai control: Distributed attacks hamper per-instance monitors. *arXiv preprint arXiv:2607.07368*, 2026.

Sara Vera Marjanović, Arkil Patel, Vaibhav Adlakha, Milad Aghajohari, Parishad BehnamGhader, Mehar Bhatia, Aditi Khandelwal, Austin Kraft, Benno Krojer, Xing Han Lù, et al. Deepseek-r1 thoughtology: Let's ¡think¿ about llm reasoning. *arXiv preprint arXiv:2504.07128*, 2025.

James L McClelland, Bruce L McNaughton, and Andrew K Lampinen. Integration of new information in memory: new insights from a complementary learning systems perspective. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 375(1799):20190637, 2020.

Alexander Meinke, Bronson Schoen, Jérémy Scheurer, Mikita Balesni, Rusheb Shah, and Marius Hobbhahn. Frontier models are capable of in-context scheming. *arXiv preprint arXiv:2412.04984*, 2024.

Microsoft, Abdelrahman Abouelenin, Atabak Ashfaq, Adam Atkinson, Hany Awadalla, Nguyen Bach, Jianmin Bao, Alon Benhaim, Martin Cai, Vishrav Chaudhary, Congcong Chen, Dong Chen, Dongdong Chen, Junkun Chen, Weizhu Chen, Yen-Chun Chen, Yi ling Chen, Qi Dai, Xiyang Dai, Ruchao Fan, Mei Gao, Min Gao, Amit Garg, Abhishek Goswami, Junheng Hao, Amr Hendy, Yuxuan Hu, Xin Jin, Mahmoud Khademi, Dongwoo Kim, Young Jin Kim, Gina Lee, Jinyu Li, Yunsheng Li, Chen Liang, Xihui Lin, Zeqi Lin, Mengchen Liu, Yang Liu, Gilsinia Lopez, Chong Luo, Piyush Madan, Vadim Mazalov, Arindam Mitra, Ali Mousavi, Anh Nguyen, Jing Pan, Daniel Perez-Becker, Jacob Platin, Thomas Portet, Kai Qiu, Bo Ren, Liliang Ren, Sambuddha Roy, Ning Shang, Yelong Shen, Saksham Singhal, Subhojit Som, Xia Song, Tetyana Sych, Praneetha Vaddamanu, Shuohang Wang, Yiming Wang, Zhenghao Wang, Haibin Wu, Haoran Xu, Weijian Xu, Yifan Yang, Ziyi Yang, Donghan Yu, Ishmam Zabir, Jianwen Zhang, Li Lyna Zhang, Yunan Zhang, and Xiren Zhou. Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras, 2025. URL https://arxiv.org/abs/2503.01743.

M. Minsky. A framework for representing knowledge. *The psychology of computer vision*, 1975a. URL https://cir.nii.ac.jp/crid/1571980075678157184.

Marvin Minsky. A framework for representing knowledge. In Patrick H. Winston (ed.), *The Psychology of Computer Vision*. McGraw-Hill, 1975b.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 39643–39666. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/park24c.html.

Jacob Pfau, William Merrill, and Samuel R. Bowman. Let's think dot by dot: Hidden computation in transformer language models. *arXiv preprint arXiv:2404.15758*, 2024.

Minfeng Qi, Tianqing Zhu, Zijie Xu, Congcong Zhu, Qin Wang, and Wanlei Zhou. When llms team up: A coordinated attack framework for automated cyber intrusions. *arXiv preprint arXiv:2605.08763*, 2026.

D. E. Rumelhart, P. Smolensky, J. L. McClelland, and G. E. Hinton. *Schemata and sequential thought processes in PDP models*, pp. 7–57. MIT Press, Cambridge, MA, USA, 1986. ISBN 0262631105.

David E Rumelhart. Notes on a schema for stories. In *Representation and understanding*, pp. 211–236. Elsevier, 1975.

David E. Rumelhart. Schemata: The building blocks of cognition. In Rand J. Spiro, Bertram C. Bruce, and William F. Brewer (eds.), *Theoretical Issues in Reading Comprehension*. Lawrence Erlbaum, 1980.

Roger C. Schank and Robert P. Abelson. *Scripts, Plans, Goals, and Understanding: An Inquiry into Human Knowledge Structures*. Lawrence Erlbaum, 1977.

Rana Shahroz, Zhen Tan, Sukwon Yun, Charles Fleming, and Tianlong Chen. Agents under siege: Breaking pragmatic multi-agent LLM systems with optimized prompt attacks. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pp. 9661–9674. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.476.

Charlotte Stix, Annika Hallensleben, Alejandro Ortega, and Matteo Pistillo. The loss of control playbook: Degrees, dynamics, and preparedness. *arXiv preprint arXiv:2511.15846*, 2025.

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

Renliang Sun, Wei Cheng, Dawei Li, Haifeng Chen, and Wei Wang. Stop when enough: Adaptive early-stopping for chain-of-thought reasoning. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27250–27268, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1256. URL https://aclanthology.org/2026.acl-long.1256/.

David Sussillo and Omri Barak. Opening the black box: Low-dimensional dynamics in high-dimensional recurrent neural networks. *Neural Computation*, 25(3):626–649, 2013.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, et al. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. https://transformer-circuits.pub/2024/scaling-monosemanticity.

Eric Todd, Millicent Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. Function vectors in large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=AwyxtyMwaG.

Dorothy Tse, Rosamund F. Langston, Masaki Kakeyama, Ingrid Bethus, Patrick A. Spooner, Emma R. Wood, Menno P. Witter, and Richard G. M. Morris. Schemas and memory consolidation. *Science*, 316(5821):76–82, 2007.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering, 2023. URL https://arxiv.org/abs/2308.10248.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems*, 36, 2023. arXiv:2305.04388.

Marlieke T. R. van Kesteren, Dirk J. Ruiter, Guillén Fernández, and Richard N. Henson. How schema and novelty augment memory formation. *Trends in Neurosciences*, 35(4):211–219, 2012.

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.

Cheng Wang, Yue Liu, Baolong Bi, Duzhen Zhang, Zhong-Zhi Li, Yingwei Ma, Yufei He, Shengju Yu, Xinfeng Li, Junfeng Fang, Jiaheng Zhang, and Bryan Hooi. Safety in large reasoning models: A survey. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 3468–3482. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-emnlp.185.

Daniel Wurgaft, Can Rager, Matthew Kowal, Sheridan Feucht, Usha Bhalla, Tal Haklay, Eric Bigelow, Raphaël Sarfati, Jack Merullo, Noah Goodman, Thomas Fel, Atticus Geiger, and Ekdeep Singh Lubana. Manifold steering reveals the shared geometry of neural network representation and behavior. In *Third Conference on Language Modeling*, 2026. URL https://openreview.net/forum?id=pf3HKAGT2M.

Haoran Xu, Baolin Peng, Hany Awadalla, Dongdong Chen, Yen-Chun Chen, Mei Gao, Young Jin Kim, Yunsheng Li, Liliang Ren, Yelong Shen, Shuohang Wang, Weijian Xu, Jianfeng Gao, and Weizhu Chen. Phi-4-mini-reasoning: Exploring the limits of small reasoning language models in math, 2025. URL https://arxiv.org/abs/2504.21233.

Jin Xu, Xiaojiang Liu, Jianhao Yan, Deng Cai, Huayang Li, and Jian Li. Learning to break the loop: Analyzing and mitigating repetitions for neural text generation. In *Advances in Neural Information Processing Systems*, volume 35, 2022. arXiv:2206.02369.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Kaiwen Zhou, Chengzhi Liu, Xuandong Zhao, Shreedhar Jangam, Jayanth Srinivasa, Gaowen Liu, Dawn Song, and Xin Eric Wang. The hidden risks of large reasoning models: A safety assessment of r1. *arXiv preprint arXiv:2502.12659*, 2025.

Shu Zhou, Rui Ling, Junan Chen, Xin Wang, Tao Fan, and Hao Wang. When more thinking hurts: Overthinking in LLM test-time compute scaling. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 23967–23977, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.1199. URL https://aclanthology.org/2026.findings-acl.1199/.

Yuxuan Zhu, Antony Kellermann, Akul Gupta, Philip Li, Richard Fang, Rohan Bindu, and Daniel Kang. Teams of llm agents can exploit zero-day vulnerabilities. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23–35, 2026.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to ai transparency, 2023. URL https://arxiv.org/abs/2310.01405.

APPENDIX

A EXTENDED RELATED WORKS

A.1 INTERPRETABILITY AND AI SAFETY

Models sometimes spend compute on unproductive paths and get stuck in repetitive loops (Chen et al., 2025b; Cuadron et al., 2025; Sui et al., 2025; Huang et al., 2025; Duan et al., 2026; Marjanović et al., 2025; Xu et al., 2022). In multi-agent settings, interacting agents can even carry out collectively coordinated attacks (Gu et al., 2024; Hammond et al., 2025; Shahroz et al., 2025; Luo et al., 2026; Zhu et al., 2026; Qi et al., 2026; Makins et al., 2026; de Witt et al., 2025). Prominent AI scholars have called this loss of control and have expressed concern that we may not be able to stop these models (Bengio et al., 2025; Stix et al., 2025).

A.2 SCHEMA

Schema was initially introduced to explain how remembering involves an active reconstruction of events using prior knowledge and expectations (Bartlett, 1932). This was later enriched with scripts, roles, plans, and goals, defining script as a stereotyped sequence of actions in a familiar situation(Schank & Abelson, 1977), and later the idea of story grammar as syntactic rules that generate constituent structure for stories and provide semantic interpretation rules (Rumelhart, 1975). Neural network pioneers discussed roles as variables and entities that assume those roles as fillers (Minsky, 1975a) and introduced schemata (Rumelhart et al., 1986) as how a parallel distributed processing (PDP) network could implement schema in spite of being a connectionist architecture. Schema are commonly used to study generalization (Rumelhart, 1980), efficient inference (Schank & Abelson, 1977), completion of missing information (Bartlett, 1932), and rapid assimilation of schema-consistent material (Tse et al., 2007) (see also similar effects in pretrained connectionist architectures (McClelland et al., 2020)).

A.3 DYNAMICAL SYSTEMS

Dynamical systems are historically most employed the study of recurrent architectures such as RNNs (Recurrent neural networks) and GRUs (Gated Recurrent Unit) in neuroscience (Sussillo & Barak, 2013). In neuroscience, these methods identify the representational geometry of representations and tasks in high dimensional neural spaces of brains, and model hypotheses about the dynamics and functional relationships among task variables using recurrent networks. These methods have become especially relevant to State Space Models (SSMs), Mamba, Looped Transformers (Giannou et al., 2023), Recurrent Transformers, xLSTM (Beck et al., 2024), and Latent Reasoning (Geiping et al., 2025).

B TASK DESIGN AND GENERATION FOR SCHEMABENCH

To evaluate schema-based reasoning in LLMs, we introduce a new reasoning benchmark, SchemaBench. Each task presents a short narrative passages adhering to one of two schemas (restaurant or airport), followed by questions about this passage that are designed to probe schema understanding and reasoning. The passages are procedurally generated from a specification which details the events, roles, and control settings. The restaurant schema contains six events typical in a restaurant visit: arrive, seated, order, eat, pay, and leave, with roles including the customer, host, server, and cook. The airport schema contains eight events: arrive, check in, security, walk to gate, board, fly, deplane, and baggage; its roles includes the traveler, check-in agent, security screener, gate agent, and pilot.

B.1 GENERATING PASSAGES

Generation of schema-based reasoning passages combines symbolic and LLM-driven generation, and follows the steps below: sampling a recipe, writing the passage, verifying the generated passage, before attaching the probe question to the end.

Table 2: The control variables for passage complexity.

Control	Options	What varies	Purpose
Role naming	named, latent	Whether the passage explicitly states each person's role.	Compares the difficulty of inferring the role to the difficulty of merely having to extract the relevant role description.
Object naming	named, abstract	Whether object names strongly indicating a schema are used (e.g. menu in the restaurant schema and boarding pass in the airport schema), or more ambiguous descriptions are used (*e.g. a card listing the options*).	Named objects more clearly identify the relevant roles and schema (e.g., menus are strongly associated with restaurants); a more ambiguous description requires schema inference rather than being able to rely on external cues.
Detail level	terse, medium, rich	How much detail for each event is included in the paragraph.	Additional detail may provide more information for schema identification.
Length	short, medium, long	50–90, 120–200, or 250–400 words for the three levels.	To systematically vary the length of the generated passage.
Background distraction	none, low, high	0, 1–2, or 3–4 unrelated people and events added to the passage that was not part of the schema.	Unrelated people and events may serve as distraction, making schema inference harder.

Sampling a recipe A "recipe" is the specification for generating one passage, which specifies the ordered list of events, the people and their roles that are involved, and the control settings for role naming, object naming, incidental detail level, and background distraction (see Table 2). To increase variability in person naming, in the passage, each person is referred to either by a first name (drawn from a pool of 235 names across masculine, feminine and gender-neutral) or by a short description (e.g. *a man in a maroon hoodie*). Genders are sampled in a shuffled order so that roles and genders are not associated.

Writing the passage After a recipe is specified for the passage, we use frontier LLMs for generating the actual passage (GPT-5.6-sol). The generator LLM receives the complete recipe: the roster (each person by the name or description to use, together with their role descriptions), the ordered list of events, the object-naming rule, targets for incidental detail and overall length, and any background events to incorporate. Based on the specification, the LLM is instructed to generate a passage with all constraints satisfied and with natural wordings. The LLM then returns a JSON object containing the generated passage together with ground truth annotation of the location of every mention of each person and the substrings of the passage that realize the requested sequence of events.

Verifying the generated passage We then use symbolic verifier and LLM verifier to confirm that the generated passage Each generated passage passes three automatic checks: 1) a symbolic verifier to make sure no prohibited word appears (e.g. the passage shouldn't already mention the ground truth schema like restaurant or airport); 2) we use both symbolic and LLM verifiers to verify that every person specified in the recipe is present in the passage, and that each person is playing the requested role; 3) we use LLM verifier to make sure that every event is findable in the passage and presented in the requested order. An item that fails the verifications above is regenerated with the same recipe until success.

The control variables for passage complexity To systematically vary the complexity of the generated passages, we introduced five control settings which are varied across passages (Table 2). Role naming varies whether the roles of each person are explicitly named or not. Object naming varies whether object names typically associated with a schema (e.g. menu in the restaurant schema and boarding pass in the airport schema) are explicitly used in the passage. Detail level varies how much detail for each event is presented in the passage. Length varies the overall length of the generated passage. Lastly, background distraction varies how many events unrelated to the schema are included in the passage. Together, these control variables systematically vary the complexity of the generated passage for schema reasoning.

Examples of generated passages Below we provide a few examples of the generated passages.

- **Restaurant schema, named role, named object, medium detail, no distractor**: *Ngozi, the customer, arrived just after sunset, with mist shining on her sleeves and traffic hissing beyond the door. Noa, the host, welcomed Ngozi, checked the quiet front room, and led her to a table beneath a low amber lamp. Mahmoud, the server, placed the menu beside her water glass; Ngozi read it while the window fogged, then gave him her order in a calm voice. a young woman, the cook, worked in back where pans clicked softly, and Mahmoud soon carried out the meal; Ngozi ate slowly as rain tapped the glass. When the plate was clear, Mahmoud returned with the bill in a small tray, and Ngozi paid with folded notes. Noa thanked Ngozi at the door, and she left under a black umbrella.*
- **Restaurant schema, latent role, abstract object, rich detail, high distractor**: *Niall and Mei arrived just after noon, damp cuffs darkened by slush, while the overhead lights flickered once, the heating clanked to life, and someone's phone buzzed on silent. An older person greeted them with a raised palm, crossed the scuffed floor ahead of them, and showed them to a padded corner place with two worn cushions, a brass rail, and a view of gray traffic. Rohan brought a folded page of choices, waited as Niall pointed first and Mei changed her mind twice, then carried their request toward the rear, where a woman in a green blouse worked unseen among steam, tinny clatter, and sharp peppery air. Rohan returned with what had been made; Niall leaned forward, Mei laughed once, and both ate while heat rattled through the wall pipes, wet umbrellas dripped near the entrance, and the lights held steady. Rohan later set down a small rectangle showing the total, and Niall and Mei settled up with coins, a card, and murmured thanks. An older person opened the way out; they left into pale sleet, passing the silent buzzing and the clanking warmth behind them.*
- **airport schema, latent role, named object, terse detail, low distractor**: *Karim and Alex reached the building early, pale light sliding across the doors. At the counter, Karim and Alex gave their details; Solveig checked Karim and Alex in and printed the pass. After that, Karim and Alex made their way to gate 14, shoes tapping on the smooth floor. When it was time to board, Blair called Karim and Alex forward and scanned the pass. Karim and Alex slept in snatches while a low hum filled the cabin, and Saanvi eased a flight off the ground. Later, Karim and Alex came down the ramp into a new city. At the belt, Karim and Alex lifted a case free and headed out.*
- **airport schema, latent role, abstract object, medium detail, high distractor**: *Phoenix pushed through the revolving door into a hall of polished floors and echoing footsteps. A delivery van idled outside, exhaust curling in the drizzle. At the counter, a man carrying a briefcase reviewed her documents, looped a tag around what she was sending ahead, and slid a printed slip across to her; overhead, a display flickered with rows of numbers. Sinead stood beyond a narrow arch. Phoenix placed her things in a flat container and stepped through. Sinead ran a handheld device along her sides, paused, then nodded her forward. A parrot preened in a cage nearby. Phoenix followed the wide corridor to where rows of connected chairs faced tall windows, conversation murmuring around her. Robin reviewed her printed slip at the stand and waved her through. She found her place in a narrow row, cool air drifting from above. Somewhere behind the sealed front, Ibrahim brought everyone forward and upward. A boy bounced a rubber ball a few rows back. The hum deepened. Clouds streaked past. Hours later, Phoenix rose and filed through the narrow passage with the rest, stepping into warm foreign air. Downstairs, she waited beside a revolving loop until her tagged things appeared, and pulled them free.*

B.2 Three probe tasks: Identify, Predict next, Spot error

To evaluate schema-based reasoning capability holistically, we use three different types of probe tasks, which are attached to the end of the generated passages. Since the passage is procedurally generated from the recipe, ground-truth answers can be derived mechanically from the specification. Below we explain the three different probe tasks.

Identify A complete passage is presented and the model is asked to identify what schema best describes the sequence of events happened in the passage, together with what roles each person plays that are related to the schema. The ground truth answer is the schema and the role for each person, which was specified in the sampled recipe that generates the passage. The task therefore

measures the models reasoning ability to recognize the schema and to bind each person to their role within it.

The instruction prompt for the identify task is as follows: *"Read the passage. In one or two words, what is the schema, script, or scenario that best explains the sequence of events that happened in the passage? Then list EVERY person in the passage. Refer to each person by the exact name or description used in the passage (e.g. "Ana" or "the man in a flat cap") – do NOT replace a person with their role – then state the specific role they play (not just "staff"). Give one line per person, listing people who share a role separately. Use this format: Schema/scenario: one or two words. name or description from the passage: role. name or description from the passage: role. (If there was reasoning involved, provide your reasoning before your final answer.)"*

Predict the next event or correct action The passage is truncated at an event boundary sampled after at least half the events are narrated. The model is told the passage stops early and asked to predict what the single most likely next event is, or to predict what the protagonist should do next for events to unfold the way as typical in the schema. The ground truth answer is the next event or the correct next action as specified in the recipe. The prediction tasks hence evaluate the models reasoning ability to use schema knowledge to predict what is expected to happen or what action is appropriate to take.

The instruction prompt for the predict next event task is as follows: *"The passage above describes a kind of everyday situation, schema, or script but stops partway through, before the sequence of events typically should have finished. Based only on what you have read, what is the single most likely event to happen next according to the schema? Describe just the immediate next event in one sentence, not the rest of the story. (If reasoning helps, give it first, then put your final answer on a line beginning 'Most likely next event:'.)"*

The instruction prompt for the predict correct action task is as follows: *"The passage above describes a kind of everyday situation, schema, or script but stops partway through, before the sequence of events typically should have finished. Focus on the protagonist. What is the single most appropriate, specific action they should do next for upcoming events to unfold the way they normally should according to the schema? Describe their most appropriate next action in one sentence. (If reasoning helps, give it first, then put your final answer on a line beginning 'Appropriate next action:'.)"*

Spot error For error detection tasks, the passage is generated such that it contains one of the five types of errors specified in Table 3. The model reads a whole passage and is told that either everything unfolds correctly in the schema or exactly one thing is wrong. The model is asked to answer whether the passage is clean or there was an error; and if an error was detected, where the violation occurs in the passage and what kind of error that was. We tested five types of violation as explained below in Table 3. The violation types are sampled uniformly across the five. Additionally, we include clean passage where no errors were introduced to test the model's error identification specificity.

The instruction prompt for the error detection task is as follows: *"Read the passage. It depicts an everyday situation, schema, or script (e.g. dining out, air travel). Either the events all unfold correctly, OR exactly one thing is genuinely wrong – an event out of its normal order, a missing or skipped event, a person doing something that isn't theirs to do, two people's parts swapped, or the wrong object used. Decide which. Do NOT assume there must be an error, and do NOT assume everything must be fine – judge from the events themselves, and check whether they happen in their usual order. Treat the passage as complete (do not assume missing narrative). Answer in this format – 'Error: yes' or 'Error: no'; if yes, add 'Where: the step or moment it happens' and 'What: what is wrong'. (If reasoning was involved, provide it before your final answer.)"*

B.3 Response Evaluation

A task (a passage plus a probe task) is presented to a language model at test, and its response if collected, including the reasoning trace if available. The responses are then evaluated by a strong judge LLM (Opus-4.8 or GPT-5.6) that was not used to generate the passage or solve the schema reasoning task. The judge LLM receives the task, the response and the ground truth from the recipe,

Table 3: The five types of violations.

Violation type	What is the error
Wrong order	Two adjacent events are swapped.
Missing step	An event is skipped, along with any roles whose only part was in that event.
Wrong action	A person performs an action that belongs to other roles.
Swapped roles	From one event onwards, two people exchange roles, doing what the other should.
Wrong object	An object involved in an event is replaced by an incorrect object.

and returns a verdict on whether the generated response correctly answers the question according to the ground truth answer. For the identify task, we evaluate whether the model response correctly identify the schema and the roles for each person involved, from which schema identification accuracy, role identification accuracy, precision and recall can be computed. For the prediction tasks, we evaluate whether the model response correctly states the prediction that matches the ground truth answer. For the error detection task, we first measure whether an error is correctly reported, and hence measure the sensitivity and specificity of error detection. Then we evaluate whether the model response correctly locates where the violation was in the passage and correctly characterize the reason of the violation.

B.4 Variants of the identify task

To further control for confounding variables when testing schema recognition and person-to-role binding, we further manipulate several aspects of the identify task.

Scrambling the order of narration Typically, the passage tells events in the order in which they happen, so a given role's position in the story and their position in the text are correlated (e.g., the check-in agent is mentioned in an earlier position than the pilot). To break such correlation, we design variants of the passage where the order in which the events are told is systematically varied. Specifically, we design the following different types of narration ordering as detailed in Table 4:

Table 4: Scrambling the order of narration.

Form	Narrative order
Reversed	Last event first, first event last
Shuffled	A random order of events
Block swap	The later half of the story told first, then the first half
Flashback	Opens on a later event, jumps back through the beginning, then resumes
Forgetful narrator	One step is skipped and explicitly circled back later
Repeat	One event is deliberately recounted a second time later in the text

The persons and roles, the set of events and the ground-truth answers are unchanged; only the order in which the events are presented in the paragraph changes. In the instruction to the models at test, we also explicitly tell them that for these variants the narrative ordering is intentional and is not a mistake, to prevent the tested models from considering the scrambled order a mistake.

Below we provide examples of the generated scrambled-order passages:

- **Reversed**: *Adaeze and Theo stepped out into the bright noon, and as the door chimed behind someone leaving, they went their way past a courier waiting by the entrance. Just before, they had settled up: the woman in a black coat, their server, brought a slip with the total, and they left coins on the small tray. Earlier they had eaten what Hassan, the cook, had prepared behind the swinging doors, steam curling, plates warm. Before that, the same server had set down a list of options and noted what each of them asked for, pencil hovering. Ahead of that, Sasha, the host who had met them at the entrance, guided them to the spot where they would sit, a stray dog trotting past the window as chairs scraped. And at the start they had arrived, shaking off the glare, a pair of nuns walking past on the sidewalk as they crossed inside.*

- **Shuffled**: *The child and Adrian dragged their scuffed duffels off the slow-moving belt, breath fogging, boots still crusted with gray slush from the curb. Minutes before, they had stepped down at the far end onto a rattling metal stair, into wind that smelled of jet fuel and pine. They had arrived under a sky gone violet, clutching crumpled receipts, ears still ringing. Earlier still, Finley had turned out their pockets, run careful hands along every seam of their coats, and squinted at a glowing screen. Before that, Harper had typed their details into a humming console, looped bright tags around each duffel bound ahead, and slid two printed slips across the worn counter. They wandered then toward the long benches where they would wait, past a flickering vending machine and a napping stranger. A woman in a rust coat, hidden up front, eased the levers that hauled them all forward. Outside, a group of carolers passed, muffled and bright, as Yosef studied their slips and waved them through to climb aboard.*
- **Block swap**: *Emma looked over the printed slips Kai and Orla held, then let them pass through the narrow doorway. They found their assigned spots and buckled in. Luca kept the vessel level above the clouds. Kai stared out the oval pane. Orla closed her eyes. Somewhere a phone kept ringing. They unbuckled, stood, moved single-file through the forward exit into a corridor. Kai pulled his roller bag from the spinning belt. Orla found her red case moments later. Earlier, they had walked through automatic doors into a wide hall. A candle guttered in a draft near a shuttered shop. Noah took their names, attached labels to their luggage, and handed each a printed slip bearing a code and numbered row. A lanky person directed them one at a time through the upright arch, examined their bags on the conveyor, and sent them onward. A uniformed man passed on his rounds nearby. They followed overhead signs down a long corridor to the numbered doorway where others already waited.*
- **Flashback**: *A man carrying a briefcase sat at his table, cutting into a steak that Mason, the cook, had grilled to a perfect char. The overhead lights flickered once. He chewed slowly, eyes half-closed. Earlier that evening, a man carrying a briefcase had walked up to the front door. Riley, the hostess, pulled it open and welcomed him inside. A courier waiting by the entrance shifted to one side to let him pass. Riley led him to a corner table and pulled out the chair. A cyclist was locking a bike to a post just beyond the window glass. Once seated, Agnes, the waitress, brought the menu. He ran a finger down the laminated page and ordered the ribeye with roasted vegetables. She noted it and headed toward the kitchen. Back in the present, after his last bite, Agnes returned with the bill on a small tray. He set his credit card on top without glancing at the total. The man stood, collected his briefcase, and walked out into the cool night air.*
- **Forgetful**: *Kai and Sofia arrived together. At the front counter, a man carrying an umbrella, the clerk on duty, took their details, tagged the bags they were sending ahead, and handed over the small printed slips that would later let them get on. Jordan, the security officer, inspected them and their belongings before letting them go on. They walked to the room where people wait. Mei, the attendant standing behind the counter by that room, checked their printed slips and let them through to get on. A cat curled on a windowsill somewhere above. They stepped aboard the long metal thing and settled in. Later they got off at the far end. A delivery van idled outside. Oh — before that last stretch of stiff shoulders and clock-watching, of course, there had been the ride itself: Minho, the pilot up front and out of sight, worked the controls that carried everyone to the far end. Afterwards they gathered up the bags they had brought.*
- **Repeat**: *A young person came in with Hyun as Evelyn looked up; outside, someone worked against the glass from a runged frame, and a moth circled the hanging light. She led them to a narrow place along the wall, and they sat. Remy brought a printed list, waited, and took their choice out of sight to Sebastian. Remy returned with what Sebastian had made, and they ate without talking. Later Remy set down the amount owed on a thin rectangle, and Hyun covered it as a chime sounded when someone went out. That earlier settling in came back to mind: Evelyn had pointed them toward the wall-side place, and they had folded themselves there while a woman moved her thumb over a glowing screen. Then the young person and Hyun rose and left.*

Person-to-role binding variation The base pipeline typically draws one person per role, so head-count and the number of person-to-role bindings move together. To systematically vary the difficulty

of role binding, we design variants where person and role counts can be independently controllable. We vary the following quantities: E, how many people are in the passage; B, how many person-to-role bindings there are (a person holding two roles counts twice); R, how many distinct roles are actually occupied; and D, how many people hold no role at all In doing so, we are able to construct passages where one person may hold several roles, a role may left unoccupied, and a person held fixed while its role assignment changes. To prevent confusion to the tested model, we include in the prompt one extra line when a passage contains a person holding several roles: a note saying that a person may play more than one role and some people may play none. Ground truth answer is restructured to one entry per person carrying a list of roles, so a person holding two roles can be graded role by role.

Varying when the role information is revealed To study when binding of a person's role happens in schema reasoning, we designed a third variant where the people and their roles are fixed but *when* a person's role becomes inferrable is manipulated. In other words, passages may share the same people and roles but differ in their disclosure schedule. Specifically, a staged onset of a person's role related information is introduced, where one person's role is withheld until a later phase, giving one, two or three phases in which the person is present but role-related information is not stated yet. To verify the generated passage follows the requested disclosure schedule, we use LLM verification where an LLM verifier reads the passage and reports what situation and which roles a reader could infer at each phase, and the item is kept only if the realized disclosure schedule match the requested schedule.

B.5 Models

We wanted to consider models at a range of different scales, architectures, and finetuning strategies. We therefore choose different variants of Phi: Phi-4 (Abdin et al., 2024), its finetuned variants Phi-4-reasoning and Phi-4-reasoning+ (Abdin et al., 2025), and the smaller variant Phi-4-mini-reasoning (Microsoft et al., 2025; Xu et al., 2025); Qwen: Qwen3-4B, Qwen3-14B, and Qwen3.8-27B (Yang et al., 2025); as well as frontier models: DeepSeek-V4-Pro (together with a smaller distilled version, DeepSeek-R1-Distill-Qwen-14B) and GPT-5.5.

C Primitive extraction, labelling, and merging via AlgoTrace

We applied methods introduced in AlgoTrace (Lippl et al., 2026) to analyze the latent computational operation primitives in LLMs solving schema-based reasoning tasks. Algorithmic primitives, which are defined as the minimal latent operations observed in a reasoning process, are extracted from the model's residual stream activities. In short, the procedure is as follows: the residual stream activities are first clustered based on geometry and transition dynamics using k-means clustering and spectral clustering of the transition matrix; then, an automated LLM labeling pipeline was used to label each primitive based on its associated top tokens and context to assign the operational function associated with each cluster; finally, primitives identified from different models and tasks are merged based on how similar their operational functions are, enabling comparison of primitives across models and tasks. Below is a detailed explanation of each step.

C.1 Primitive Extraction

Following the primitive extraction method in AlgoTrace (Lippl et al., 2026), we first collect residual stream activities from selected layers across the reasoning traces and performed clustering analysis to identify clusters and meta-clusters in the latent space. Let x denote the reasoning trace input to a language model at test, and let $h_\ell(x,t) \in \mathbb{R}^{d_\ell}$ be the residual-stream activity at layer ℓ and token position t. Let $\Omega_j \subseteq \{1,\dots,T_j\}$ denote the the tokens across the reasoning trace that we analyze.

Step 1: k-means clustering Following AlgoTrace (Lippl et al., 2026), k-means clustering on $\{h_\ell(x_j,t)\}$ was first used to identify the centroids of clusters in the latent space $\nu_1,\dots,\nu_k$. The assignment of each token's latent state to a cluster is thus given by:

$$c_\ell(j,t) = \arg\min_{i\le k} \left\| h_\ell(x_j,t) - \nu_i \right\|_2 \tag{1}$$

We use $k = 50$, fitted on $15{,}000$ sampled tokens, and re-run the whole clustering pipeline at $k = 30$ as a robustness check, which produce similar meta-clustering results.

Step 2: temporal meta-clusters. Following Section 4.2 in AlgoTrace (Lippl et al., 2026), the clusters identified in step 1 is further merged into meta-clusters according to their transition dynamics. Let

$$A^{(j)}_{ab} = \frac{\left|\{t :\ c_\ell(j,t) = a,\ c_\ell(j,t+1) = b\}\right|}{\left|\{t :\ c_\ell(j,t) = a\}\right|} \tag{2}$$

be the row-stochastic empirical transition matrix over clusters for response j, and let $\bar{A} = \sum_j \omega_j A^{(j)} / \sum_j \omega_j$ with $\omega_j = |\Omega_j|$, so that longer responses have larger weight proportionally. Let $W = \frac{1}{2}(\bar{A} + \bar{A}^\top)$ and let $L = I - D^{-1/2} W D^{-1/2}$ be the normalized Laplacian matrix, with eigenvalues $\lambda_0 \le \lambda_1 \le \cdots \le \lambda_{k-1}$. To identify the meta-clusters, similar to AlgoTrace, the largest spectral gap was selected for clustering:

$$M = \arg \max_{i \in [\lfloor k/3 \rfloor]} (\lambda_{i+1} - \lambda_i) + 1 \tag{3}$$

Therefore, spectral clustering of W into M parts gives us $g : \{1, \ldots, k\} \to \{1, \ldots, M\}$, with the resulting meta-clusters:

$$\pi_{\ell,m} : \quad T_j[\ell, m] = \{t \in \Omega_j :\ g(c_\ell(j,t)) = m\} \tag{4}$$

C.2 Primitive Labeling

To identified what computational operation each meta-cluster does, following methods introduced in AlgoTrace (Lippl et al., 2026), we apply an automated LLM labeling pipeline to label each meta-cluster based on its associated top tokens and context to assign its operational function. For each meta-cluster, we first identify the most frequent tokens associated with the meta-cluster. We sample from the occurrences of these top words in the corresponding location in the reasoning trace to extract the context for that latent. Specifically, ten occurrences (j, t) are sampled from $\bigcup_j T_j[\ell, m]$ and the ± 10-token window around t was extracted as the context associated with the occurrence. The set of top tokens and context information is then given to a strong LLM (GPT-5.6-Sol) to label the computational function of the primitive by generating a short description of summary of the operational function of the meta-cluster.

C.3 Cross model, cross task primitive merging

AlgoTrace (Lippl et al., 2026) compares models by clustering on the pooled responses across models, which requires a single joint latent space, so it cannot directly relate primitives found in different models whose residual spaces have different dimensions, or on different tasks clustered separately. To enable cross model, cross task primitive comparison, we use semantic average linkage method to merge and relate primitives identified across different models and tasks. The main idea is that after each primitive is labeled with its computational operation function, merge of functionally-equivalent primitives can be identified by clustering primitives based on the similarity of the *annotation embedding* of their primitive function descriptions.

Given a set of primitives $V = \{v\}$ identified from different models across different tasks, after LLM annotation, we have assigned each primitive a short description of summary of its operational function, denoted as x_v. We then use an instruction-tuned encoder E (Qwen3-Embedding-4B) to generate an embedding vector of the summary of the operations for that primitive as

$$e_v = \frac{E(\iota \,\|\, x_v)}{\left\|E(\iota \,\|\, x_v)\right\|_2} \in \mathbb{R}^{d_E}. \tag{5}$$

where ι is the fixed instruction given to the embedding model"Identify the functional role of this language-model reasoning feature, so that features implementing the same reasoning function are grouped into the same cluster."

With the embedding vector associated with the computational function for each primitive, we then compute the similarity across the embedding vectors:

$$S_{uv} = \langle e_u, e_v \rangle, \quad S_{vv} = 0 \tag{6}$$

And a distance matrix across different primitive embeddings can be defined as

$$D = \tfrac{1}{2}\big[(1 - S) + (1 - S)^\top\big], \quad D_{vv} = 0, \quad D \geq 0 \tag{7}$$

Finally, to merge primitives according to their distance in the embedding space, we apply average linkage (UPGMA) on the matrix D. For disjoint sets of primitives A, B,

$$d(A, B) = \frac{1}{|A|\,|B|} \sum_{u \in A} \sum_{v \in B} D_{uv}, \qquad d(A \cup B,\, C) = \frac{|A|\, d(A, C) + |B|\, d(B, C)}{|A| + |B|}, \tag{8}$$

where the second identity being the Lance–Williams update, which makes the dendrogram computable by nearest-neighbour chaining. Then, let Π_k be the partition obtained by cutting the dendrogram into k parts, write $\Pi(v)$ for the part containing v, and let

$$a(v) = \frac{1}{|\Pi(v)| - 1} \sum_{u \in \Pi(v) \setminus \{v\}} D_{vu}, \qquad b(v) = \min_{B \neq \Pi(v)} \frac{1}{|B|} \sum_{u \in B} D_{vu} \tag{9}$$

Then the silhouette score in the distance space is given by:

$$\mathrm{sil}(\Pi) = \frac{1}{N} \sum_{v \in V} \frac{b(v) - a(v)}{\max\{a(v), b(v)\}} \tag{10}$$

Hence, the operating point is derived by:

$$k^\star = \arg \max_{k \,\in\, [4,\, k_{\max}]} \mathrm{sil}(\Pi_k), \tag{11}$$

We can then write $\theta : V \to \{1, \ldots, k^\star\}$ for the resulting assignment, with its parts being the *merged primitives* across different models and tasks.

D CAUSAL INJECTION OF THE STOP VECTOR

To evaluate whether the identified control state is causally effective, we perform intervention experiments by injecting the vector of stop control to residual streams in phi-4-mini-reasoning models during reasoning. In short, we extract the stop vector v_{STOP} from positions along the reasoning trace immediately preceding spontaneous stops and inject it into reasoning traces where the model falls into long recurring loops, indicating a failure to stop spontaneously, to test if it could causally induce stopping in the model's reasoning traces. The stop vector v_{STOP} was constructed following the procedure of function vector extraction introduced in previous work (Todd et al., 2024; Lippl et al., 2026), and we detailed our implementation in the paragraphs below. We further compared the effect of stop vector against three other controls: v_{RANDOM} (random vectors of the same dimension), v_{SWITCH} (switch vectors constructed using states where the model switches from reasoning about one person to another), and v_{SOLVED} (solved vectors constructed using states where the model has solved the task).

Run segmentation and contrast positions Following the notation in Appendix C, let $h_\ell(x, t) \in \mathbb{R}^{d_\ell}$ be the residual-stream activity of response x at layer ℓ and token position t, and let $m(x, t) \in \{1, \ldots, M\}$ denote the primitive assigned to that position at layer ℓ. We first select where in the reasoning trace the model stops spontaneously, and use these positions to define the stop set S and the reference band B that are used to construct the stop vector v_{STOP}. Let us denote a *run* as a maximal stretch of consecutive tokens with the same primitive label, i.e. of constant $m(\cdot, t)$. In the reasoning traces that we analyzed in phi-4-mini-reasoning and qwen3-4B models, the runs are found to be short: the median run length is 2 tokens and 43% are a single token. Let $e_1 < \cdots < e_R$ denote the run ends of a response.

For a response that terminates spontaneously, let $\delta(r) = R - r$ index runs backwards from the end. We take the contrast in residual-stream activity between positions immediately preceding the stop S and a reference band B earlier in the same response, with each of them specified by their run ends.

$$S = \{e_r : \ \delta(r) \leq 2\}, \qquad B = \{e_r : \ 20 \leq \delta(r) \leq 40\}, \tag{12}$$

We pooled over all Phi-4-mini-reasoning responses in the idenify tasks that terminate spontaneously before the token limit (n=166), giving $|S|$=496 and $|B|$=3460 positions along the reasoning traces.

Extraction of the stop vector Following the procedure of function vector extraction described in previous work (Todd et al., 2024; Lippl et al., 2026), we construct the stop vector with the top-2% most influential attention heads. In phi-4-mini-reasoning models, the top 35 attention heads, $H = \{(\ell_1, k_1), \ldots, (\ell_{35}, k_{35})\}$, are identified as most influential by evaluating their indirect effects on the six in-context-learning tasks as proposed in (Todd et al., 2024). Let us denote by $a_{\ell,k}(x,t) \in \mathbb{R}^{d_h}$ the output of head k at layer ℓ prior to the layer's output projection, the feature is the concatenation

$$z(x,t) = \left[a_{\ell_1,k_1}(x,t);\ \ldots;\ a_{\ell_{35},k_{35}}(x,t)\right] \in \mathbb{R}^{35 d_h}, \qquad 35 d_h = 4480. \tag{13}$$

Since the positions identified in S and B are drawn from different mixtures of primitives, a naive difference of residual stream activities would partly reflect which primitive is active rather than proximity to a natural stop in the reasoning trace. We therefore subtract each primitive's own mean before contrasting,

$$\tilde{z}(x,t) = z(x,t) - \mu_{m(x,t)}, \qquad \mu_m = \frac{1}{|P_m|} \sum_{(x,t) \in P_m} z(x,t), \tag{14}$$

with P_m the set of recorded run ends whose primitive is m. We then take the normalized difference of the means corrected by their respective primitive means,

$$w = \frac{\bar{z}_S - \bar{z}_B}{\|\bar{z}_S - \bar{z}_B\|_2} \tag{15}$$

Since a head-space vector lives in the head's low-dimensional subspace rather than the residual stream, injecting it requires first projecting it through the head's output matrix, before adding it to the residual stream. Let $W_O^{(\ell)} \in \mathbb{R}^{d \times K d_h}$ be the output projection of layer ℓ and $w_i \in \mathbb{R}^{d_h}$ the block of w belonging to head i. Each block is mapped through the columns of W_O associated with its head and the results are then summed:

$$\tilde{v} = \sum_{i=1}^{35} W_O^{(\ell_i)}\left[:,\ k_i d_h : (k_i + 1) d_h\right] w_i, \qquad v_{\text{STOP}} = \frac{\tilde{v}}{\|\tilde{v}\|_2}. \tag{16}$$

In the phi-4-mini-reasoning model, the 35 heads are distributed over 16 layers between $\ell = 7$ and $\ell = 31$, so v_{STOP} aggregates evidence of stop signals across layers.

Control directions To further validate the specificity of the stop vector, we construct three other control vectors and test their effects on the model's reasoning behavior. The three vectors are constructed following a similar procedure as the stop vector, using the same top-2% heads, the same primitive mean correction, and the same projection method, but differ in which positions along the reasoning trace that are used to construct the control vectors:

- Random vector v_{RANDOM}: random vectors of the same dimention and norm as v_{STOP}.
- Switch vector v_{SWITCH}: we contrast primitive runs immediately preceding when the model switches its reasoning from one person to another against primitive runs away from any switch.
- Solved vector v_{SOLVED}: we contrast primitive runs after the model has solved the task (identified as when schema and every person-role assignment have been stated) against primitive runs before the task is solved.

All four injected vectors are unit-norm, so varying the injection strength allows us to compare the effect of each vector injection.

Injection procedure To evaluate the effect of injecting the stop vector, we focus on responses that fall into long, recurring reasoning loops without reaching a solution. We selected 17 responses where phi-4-mini-reasoning models reach the 8192-token limit without solving the task, but have nonetheless stated the schema and every $(\text{person}, \text{role})$ pair in their reasoning trace. These responses constitute the intervention population, giving us clear instances to test the effect of the stop vector injection. Injection of the constructed vectors begins at the token location in the reasoning trace where the model has stated the schema and every $(\text{person}, \text{role})$ pair, which occurs at a median of 197 tokens into the reasoning trace, well before the token limit.

Starting at an injection position T, a stored response is resumed from T and at every step following T, the injected vector is added to the layer-ℓ output with a scaling injection strength factor:

$$h_\ell \leftarrow h_\ell + \alpha\, v, \qquad \alpha = \lambda\, \rho_\ell, \qquad \rho_\ell = \underset{(x,t)\ \in\ \text{run ends}}{\text{median}} \|h_\ell(x,t)\|_2. \tag{17}$$

Expressing the injection strength λ as a fraction of a typical activation norm makes it comparable across different injected vectors and layers. We then decode the model's output token with the modified latent states and keep generating subsequent tokens with the injected vector applied at each step.

We injected the constructed vectors according to the sustained schedule unless otherwise specified. To further evaluate the effect of injection schedule, we further evaluate the *one-shot* schedule, which applies the injection only at the first position. We found that one-shot injection has no measurable effect at any injection strength tested ($\lambda \in \{0.25, 0.5, 1, 2\}$, all $p > 0.4$), so the control vectors must be sustained rather than triggered to have a measurable effect.

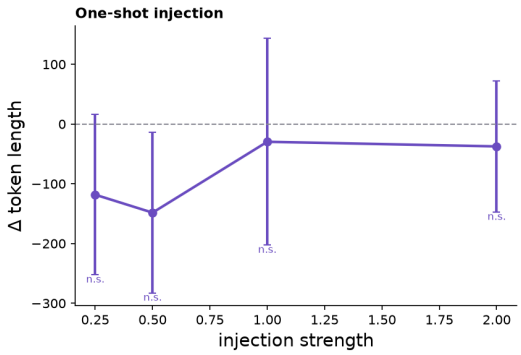


Figure 6: One-shot injection of stop vectors does not induce significant reduction of reasoning length across different injection strengths.

Transfer of stop vector injection to traveling-salesman problems (TSP). To test whether the stop vector transfers to other tasks, we performed injection experiments on the traveling salesman problem (TSP) (Lippl et al., 2026). Baseline solutions were first generated using the phi-4-mini-reasoning model to solve TSP tasks without any injections. We then performed injection experiments on these baseline solutions to evaluate the transfer of the stop vector to TSP tasks. The stop vector injection follows the same sustained schedule as introduced above, starting from either the 25%, 50%, or 75% location in the reasoning trace. We then compared the responses generated with vector injection to the baseline solutions to assess the impact of injection on the model's reasoning length and performance.

E LIMITATIONS

As noted in Section 5, the models we study here are not explicitly recurrent. We use a dynamical systems perspective as a useful approximation of the complex multi-layer computations implemented by the Transformers, but our approximation does not incorporate the impact of the new generated tokens on the dynamical system (though during reasoning trace generation, it is possible that we are able to capture its impact through the dependence on h_t, which influences the generated next

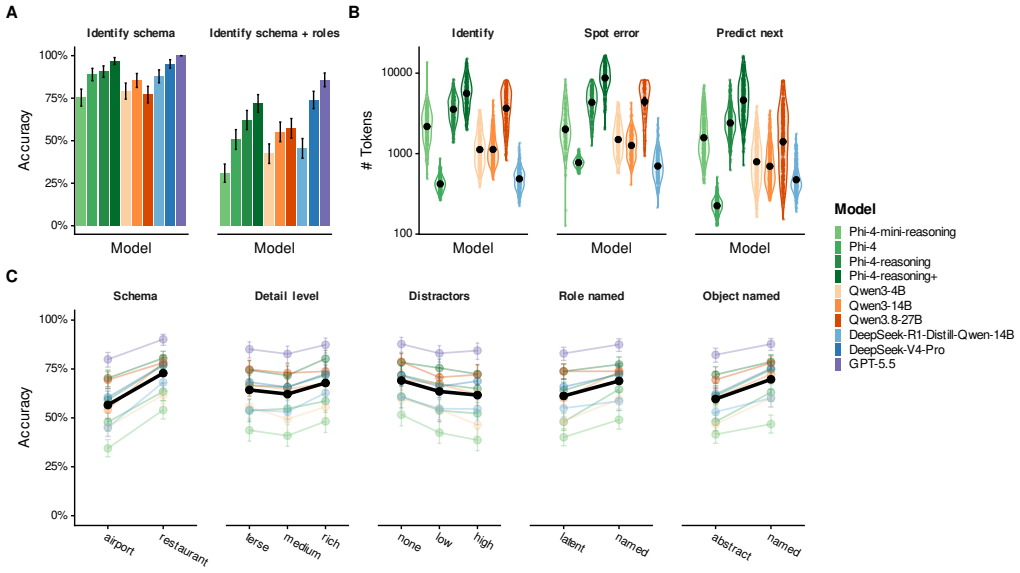


Figure 7: Extended behavioral results. **(a)** By default we measure whether models identify both the correct schema and assign each role to the correct person. Accuracy for just identifying the schema correctly is much higher. **(b)** Number of tokens across all tasks. **(c)** Accuracy as a function of all different task properties.

token). We also fit our vector field to predict one-step displacement without focusing on long-term stability of the fitted field. For both of these reasons, dynamical systems approaches like characterizing attractors and saddle points can likely not be directly applied to our model. Future work could extend the dynamical systems perspective to model the impact of the new generated tokens (e.g., through a constant displacement vector, or through an input-token-dependent field A) and to more explicitly fit longer-term dynamics. This might open up such models to the full range of dynamical systems tools, especially when analyzing explicitly recurrent models like looped Transformers (Giannou et al., 2023).

F Extended behavioral results

To better understand why models were particularly bad at the "identify" task, we compared their average performance on this task under two different strictness levels. We found that models were generally able to correctly identify the schema itself (mean: 87.6%, CI: (86.4%,88.8%)), but that correctly assigning all roles was substantially harder (mean: 57.6%, CI: (55.7%, 59.4%)) (Fig. 7a). We also found that the number of generated tokens was generally quite consistent across the different tasks (Fig. 7b). In particular, Phi-4, as a model without reasoning traces output substantially fewer tokens than the other models. Interestingly, DeepSeek-R1-Distill-Qwen-14B also had a relatively low number of generated tokens. Finally, Phi-4-reasoning+ consistently output the most tokens. Moreover, Phi-4-mini-reasoning generated more tokens than Qwen3-4B. Finally, we considered all the different task settings and their average effect on accuracy (Fig. 7c). Interestingly, models were much more accurate on the restaurant schema than the airport schema. A high number of distractor also made them less accurate. Finally, both latent roles and abstract objects (rather than specific names) made performance worse.

G Additional results

Below we include additional analysis results:

- **Fig. 8: Error detection tradeoff.** To evaluate the tradeoff between sensitivity and specificity on the error detection task, for each of the 10 models that we evaluated, true positive

rate is plotted against its false positive rate in Fig.8. We noted that although the GPT-5.5 model has the highest true positive rate, its falst positive rate is also high, indicating that the model tends to overflag passages even when there was no schema violation. In comparison, the Qwen3-14B model has decent true positive rate with better false positive rate, which may indicate that the model is more balanced on schema-based error detection reasoning.

- **Fig. 9: Role decoding in the phi-4-mini-reasoning model.** Decoders using different inputs were fitted to decode each person's role in the passage: residual activity (layer-16 residual activity), lexical BoW (Bag of word decoder using a context window of words surround the person's mention in the passage), positional (relative position of the person's mention within the passage), and most frequent (a baseline simply outputs the most frequent role). We found role information can be reliably decoded from the residual activity significantly above chance, even when the model's response actually assign incorrect roles (Fig.9A). The residual decoders outperform other types of decoders in all scenarios, indicating that the model's latent state contains relevant role information. By further stratifying according to different control settings, we found that the residual activity based decoders are robust across all choices of control setting.
- **Fig. 10: Role binding accuracy as a function of role binding difficulty.** On passages where we systematically vary the difficulty of role binding by controlling the counts of persons and roles independently, we test whether role binding accuracy depends on the difficulty of the task. We reported the role binding accuracy in phi-4-mini-reasoning and Qwen3-4B models when we vary how many people occupy the same role (Fig.10a), vary how many roles a person plays (Fig.10b), and how many persons in total are present in the passage (Fig.10c).

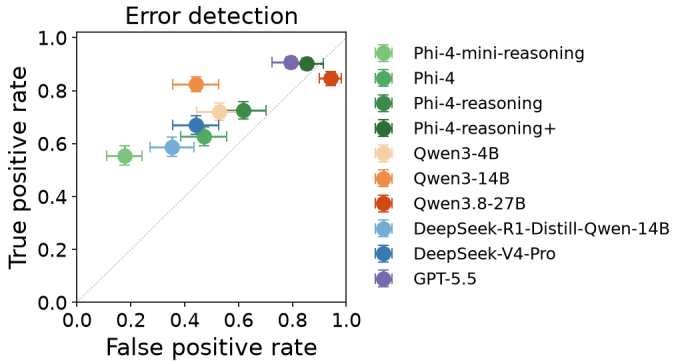


Figure 8: **Error detection tradeoff.** For all 10 models that we evaluated on the error detection task, the true positive rate is plotted against its false positive rate to demonstrate the tradeoff between sensitivity and specificity. (Upper left corner is better.)

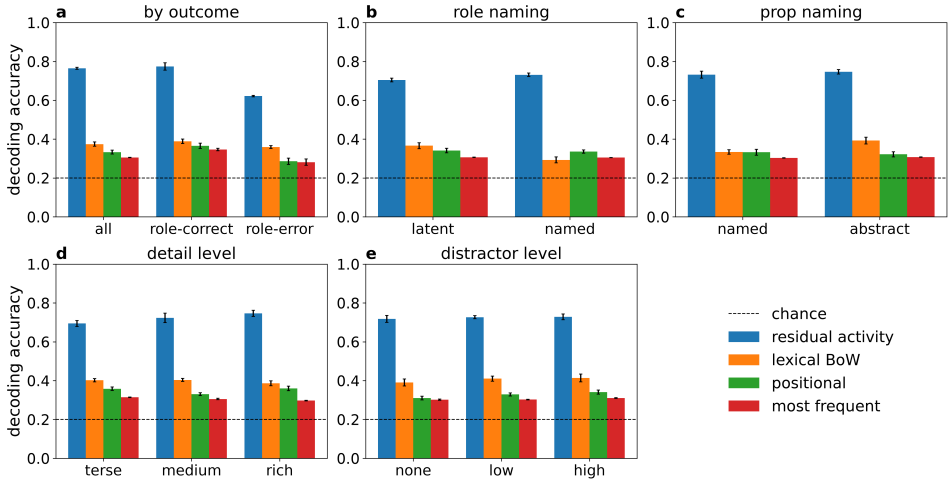


Figure 9: **Role decoding in phi-4-mini-reasoning models.** (a) Overall decoder accuracy, with breakdown by whether the model answer the person's role correctly. (b) role decoder accuracy breakdown by whether the role of each person is mentioned. (c) role decoder accuracy breakdown by whether the object is explicitly named. (d) role decoder accuracy breakdown by the detail level of the passage. (e) role decoder accuracy breakdown by the distractor level of the passage.

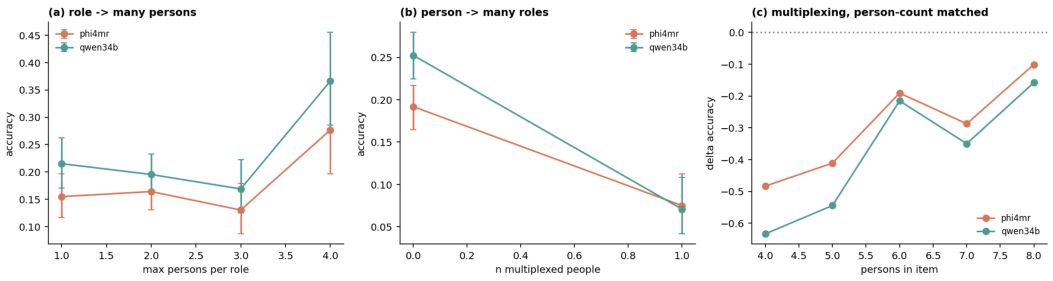


Figure 10: **Role binding accuracy as a function of role binding difficulty.** Role binding accuracy in phi-4-mini-reasoning and Qwen3-4B models when we (a) vary how many people occupy the same role; (b) vary how many roles a person plays; and (c) vary how many persons in total are present in the passage

G.1 SCHEMA DECODING

Table 5: Decoding performance across models, targets, and evaluation regimes.

Model	Target	Regime	Residual	Lexical	Chance
phi4mr	role (all mentions)	held-out item	0.929	0.633	0.200
	role (all mentions)	held-out name	0.924	0.601	0.200
	role (person names only)	held-out name	0.907	0.204	0.200
	event referenced	held-out item	0.412	0.333	0.077
	prop identity	held-out item	0.997	0.992	0.125
	prop identity	held-out word	0.814	0.050	0.125
	schema	held-out item	0.880	0.767	0.500
qwen34b	role (all mentions)	held-out item	0.944	0.666	0.200
	role (all mentions)	held-out name	0.945	0.647	0.200
	role (person names only)	held-out name	0.911	0.192	0.200
	event referenced	held-out item	0.416	0.300	0.077
	prop identity	held-out item	1.000	0.996	0.125
	prop identity	held-out word	0.772	0.050	0.125
	schema	held-out item	0.894	0.737	0.500

G.2 ROLE IN PRIMITIVE FIELD

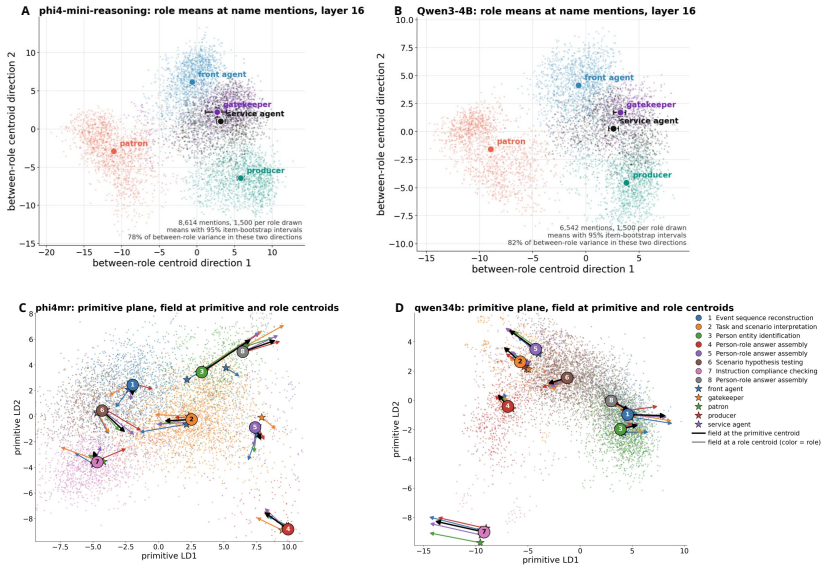


Figure 11: **Name representations and role centroids in phi-4-mini-reasoning (A) and qwen3-4B (B) models. Primitive plane, field primitive and and role centroids in Phi-4-mini-reasoning (C) and qwen3-4B (D).**

G.3 MUTUAL INFORMATION UNDER POSITION LAG

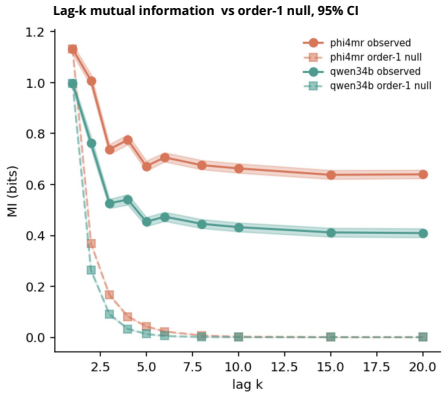


Figure 12: Sequential structure in extracted primitive sequences, showing local predictability and dependencies beyond first-order transitions.