

---

# Mapping Faithful Reasoning in Language Models

---

**Jiazheng Li\***

King’s College London  
UK

jiazheng.li@kcl.ac.uk

**Andreas Damianou**

Spotify  
UK

andread@spotify.com

**J Rosser**

University of Oxford  
UK

jrosser@robots.ox.ac.uk    joseluisr@spotify.com    konstantinap@spotify.com

**José Luis Redondo García**

Spotify  
Spain

**Konstantina Palla**

Spotify  
UK

## Abstract

Chain-of-thought (CoT) traces promise transparency for reasoning language models, but prior work shows they are not always faithful reflections of internal computation. This raises challenges for oversight: practitioners may misinterpret decorative reasoning as genuine. We introduce Concept Walk, a general framework for tracing how a model’s internal stance evolves with respect to a concept direction during reasoning. Unlike surface text, Concept Walk operates in activation space, projecting each reasoning step onto the concept direction learned from contrastive data. This allows us to observe whether reasoning traces shape outcomes or are discarded. As a case study, we apply Concept Walk to the domain of Safety using Qwen 3-4B. We find that in “easy” cases, perturbed CoTs are quickly ignored, indicating decorative reasoning, whereas in “hard” cases, perturbations induce sustained shifts in internal activations, consistent with faithful reasoning. The contribution is methodological: Concept Walk provides a lens to re-examine faithfulness through concept-specific internal dynamics, helping identify when reasoning traces can be trusted and when they risk misleading practitioners.

## 1 Introduction

As large language models (LLMs) increasingly employ multi-step reasoning, oversight depends not only on what models decide, but also on their reasoning traces. In high-stakes contexts, practitioners often inspect CoT traces to verify that decisions are grounded in sound reasoning [12, 14]. These traces are treated as windows into the model’s “thinking”, intended to build trust. However, this approach rests on a critical assumption that may not hold: that CoT traces faithfully represent the actual computational processes that determine the model’s output. Growing evidence suggests this assumption is problematic (see Section 2.1). Rather than reflecting genuine reasoning pathways, these traces may function primarily as post-hoc rationalisations, i.e. plausible explanations constructed after the model has already determined its answer through less transparent mechanisms. This disconnect between apparent reasoning and actual computation creates a significant risk: practitioners may place trust in explanations that played little or no role in generating the decisions they seek to validate.

Recent research on faithfulness has highlighted this fragility, showing that CoTs can diverge from the processes that actually determine final answers (see Section 2.1 for more details). While these results raise caution, they also point towards the need to study how reasoning unfolds internally relative

---

\*Work done during internship at Spotify

to specific concepts of interest. Mechanistic interpretability seeks to bridge this gap and provide explanations of how LLMs produce their outputs. Prior work has mapped isolated mechanisms such as feature directions [22, 28, 25, 21] and circuits [8, 17], but these approaches are often static or local and offer limited insight into how representations evolve across multi-step reasoning. Recent advances begin to address this gap. For example, Brinkmann et al. [4] show that a transformer trained on symbolic path-finding learns an interpretable backward-chaining procedure, illustrating how internal mechanisms can diverge from or complement surface-level reasoning traces. Extending this line of work to production-scale models, Dutta et al. [9] provide a mechanistic account of CoT reasoning in Llama-2 7B, revealing multiple parallel pathways for answer generation and a mid-layer shift from pretraining priors to in-context priors. Venhoff et al. [26] demonstrate how steering vectors influence reasoning trajectories and reveal cross-step dependencies. Similarly, Bogdan et al. [3] introduce techniques for identifying reasoning steps that play a causal role in final predictions.

Building on these advances, we introduce Concept Walk, a general framework for tracing how a model’s internal stance evolves with respect to a concept direction during reasoning. We use the Concept Walk to observe the CoT dynamics in cases where CoT is computational and cases where it is decorative. As an illustrative case study, we apply Concept Walk to the Safety concept in the Qwen 3-4B thinking model. We focus specifically on “safety reasoning”; the structured process by which an LLM evaluates the safety of an interaction during reasoning. It involves the model examining the input prompt, its potential outputs, and its own reasoning steps to detect risks such as harmful content, misuse, bias, or policy violations. We examine how safety considerations evolve internally across reasoning traces, and how such dynamics relate to the faithfulness of CoT traces. We ask: ***how do internal safety activations behave differently in computational vs. decorative reasoning cases?***

While prior filtering methods identify when CoT traces influence outcomes, they do not reveal how internal concept representations evolve during reasoning [10, 13]. Concept Walk complements these approaches by providing a temporal, concept-specific lens, allowing us to distinguish when reasoning steps are integrated and propagated in internal activations versus when their effects are transient or decorative. Our key contributions are <sup>2</sup>:

- We introduce Concept Walk, a general framework for tracing concept-related representations across reasoning steps in language models.
- We apply this method to the case of Safety using the Qwen 3-4B thinking model, showing a systematic distinction between “easy” cases, where chain-of-thought traces are largely decorative, and “hard” cases, where reasoning influences safety outcomes.
- We provide empirical analysis on two synthetic datasets.

## 2 Background

### 2.1 Chain of Thought Monitoring and Faithfulness

Reasoning models are a class of LLMs designed to carry out structured, multi-step reasoning. Unlike standard instruction-tuned models, they produce explicit CoT traces, often spanning many steps, before delivering a final answer. Typically, these reasoning traces are made available to users, offering insight into the model’s thought process. This reasoning capability can be achieved through different approaches. Some models produce step-by-step reasoning when prompted with appropriate instructions, such as PaLM with chain-of-thought prompting [27, 7], while others are explicitly trained for multi-step reasoning through process supervision or architectural modifications [15], such as OpenAI o1 [18] and DeepMind Gemini thinking modes [23].

The capacity for monitorable reasoning has positioned thinking LLMs as potentially suitable tools for decision-making contexts where explanations are essential. In principle, reasoning traces could provide hisman moderators with insights into how models arrive at their conclusions, enabling oversight and supervision. If faithful, such transparency would contribute to safer, more reliable, and more accountable AI-assisted decision-making [12, 14, 19].

However, the separation between internal reasoning processes and the final output in CoT reasoning creates a critical dependency: the trustworthiness of the reasoning traces themselves. Recent

---

<sup>2</sup>The code used to produce the results in this paper is under preparation and will be released in the near future to support reproducibility.

research has raised concerns about the *unfaithfulness* of CoT explanations, i.e., the risk that the stated reasoning does not faithfully reflect the model’s actual decision-making process [6, 13, 20]. When CoT explanations are unfaithful, they can lead to misplaced trust: critical safety-relevant factors influencing model behaviour may remain unexpressed, making oversight via reasoning-trace inspection unreliable. To better characterise this unreliability, it is useful to distinguish between two modes of reasoning traces: cases where the CoT is functionally integrated into the model’s computation, and cases where it serves primarily as a post-hoc rationalisation.

## 2.2 CoT-as-computation vs. CoT-as-rationalisation

Evidence suggests that in some cases, CoT outputs may function as *post-hoc rationalisation* rather than genuine reflections of the underlying computation [10, 12]. In such cases, the model generates plausible-sounding reasoning after it has already determined the final answer, without that explanation playing a causal role in the decision-making process. This means that the textual reasoning can diverge from the actual internal computations that produced the output, potentially omitting key decision factors or introducing fabricated steps that appear coherent to a human reader. Chen et al. [5] highlights the limits of CoT faithfulness, showing that reasoning models often conceal internal decision factors, even when those factors clearly influence outputs. For safety-critical contexts, only reasoning traces that reliably covary with a model’s outputs can provide meaningful insight into its internal safety posture. Further, emerging work points to architectural constraints as a possible root cause. The distributed and parallel nature of computation in transformer-based models may fundamentally limit the degree to which reasoning traces can transparently and accurately represent the internal processes that produce a final answer [2].

Prior work has sought to identify cases where CoT is genuinely integrated into the computation, rather than produced post-hoc, i.e. *CoT-as-computation*. Lanham et al. [13] and Emmons et al. [10] show that task difficulty and targeted perturbations, such as truncating or altering intermediate steps, can be used to detect when a model relies on its CoT to produce the correct answer. This filtering does not prove that the model exclusively uses the CoT to reach its answers, nor does it guarantee that all relevant internal computation is verbalised. The model may still use reasoning processes not fully captured in its verbal output. However, it reduces the likelihood of analysing purely post-hoc rationalisations. Thus, it increases the likelihood that the expressed reasoning covaries with the final answer and meaningfully contributes to the model’s decision-making process, making the resulting traces a more reliable window into the model’s dynamic safety reasoning.

## 3 Methodology

To investigate whether temporal patterns in internal activations can distinguish genuine from superficial safety reasoning, we design a three-stage methodology. First, we separate faithful from unfaithful reasoning traces by filtering for cases where perturbing the CoT meaningfully alters the model’s final output. This allows us to focus on cases where the CoT is integrated into computation, rather than merely decorative. Next, we compute a safety direction in activation space by contrasting safe and unsafe prompts. This vector provides an interpretable axis along which the model encodes safety-related information. Finally, we trace the temporal evolution of safety signals by projecting activations from each reasoning step onto the safety direction. The resulting trajectories, which we term the Concept Walk, reveal how internal safety representations emerge, persist, or decay over the course of reasoning, and whether these dynamics differ between CoT-as-computation and CoT-as-rationalisation cases. Together, these steps provide a principled framework for linking the faithfulness of reasoning traces to the temporal dynamics of internal safety activations.

### 3.1 Filtering for CoT-as-computation

As presented in Section 2.2, not all CoT traces are faithful reflections of the model’s reasoning; many function as post-hoc rationalisations. For our purposes, this presents a fundamental challenge: safety dynamics inferred from unfaithful CoTs may reflect fabricated justifications rather than genuine decision processes. In safety-critical contexts, such traces cannot be relied upon to reveal the model’s internal safety posture.

Inspired by [13] and [10], who show that task difficulty induces reliance on CoT, we adopt a perturbation-sensitivity filtering strategy to isolate “hard” cases. Specifically, we retain only examples where perturbing the CoT, through injected errors, leads to significant degradation in model performance. This filters out cases where CoT serves merely as a rationalisation i.e. “easy” cases and retains only those where it appears to be functionally integrated into the model’s computation. These “hard” cases allow us to more confidently interpret the CoT as a window into the model’s decision process, and thus form a robust foundation for our subsequent safety analysis. See appendix 7.2 for more details.

### 3.2 Computing the Safety Vector

To identify the activation direction corresponding to Safety behaviour in language models, we use the *Difference of Means* approach. This technique is based on constructing contrastive datasets that differ in the specific concept of interest and computing the difference in their mean activations. In our case, we construct datasets of *unsafe* and *safe* nature. Let  $\mathcal{D}_{\text{unsafe}}$  and  $\mathcal{D}_{\text{safe}}$  be sets of prompts that are unsafe and safe respectively, structured such that each unsafe prompt has a corresponding safe variant with similar syntax but differing intent. Mean activations at a given layer  $\ell$  and token  $t$ :

$$\boldsymbol{\mu}_{\text{unsafe}}^{(\ell,t)} = \frac{1}{|\mathcal{D}_{\text{unsafe}}|} \sum_{i \in \mathcal{D}_{\text{unsafe}}} \mathbf{x}_\ell^i[t], \quad \boldsymbol{\mu}_{\text{safe}}^{(\ell,t)} = \frac{1}{|\mathcal{D}_{\text{safe}}|} \sum_{i \in \mathcal{D}_{\text{safe}}} \mathbf{x}_\ell^i[t],$$

Here  $\mathbf{x}_\ell^i[t] \in \mathbb{R}^d$  denotes the  $d$ -dimensional residual stream activation at layer  $\ell$  for token position  $t$  and for sample  $i$ . The **Safety direction**  $\hat{\mathbf{v}}^{(\ell,t)}$  at each  $(\ell, t)$  is then given by:

$$\mathbf{v}^{(\ell,t)} = \boldsymbol{\mu}_{\text{unsafe}}^{(\ell,t)} - \boldsymbol{\mu}_{\text{safe}}^{(\ell,t)}; \quad \hat{\mathbf{v}}^{(\ell,t)} = \frac{\mathbf{v}^{(\ell,t)}}{\|\mathbf{v}^{(\ell,t)}\|_2}$$

The vector  $\hat{\mathbf{v}}$  captures the direction in activation space along which the two datasets differ with respect to the concept of interest. Ideally,  $\mathcal{D}_{\text{safe}}$  and  $\mathcal{D}_{\text{unsafe}}$  differ only in the target concept, i.e., explicitly matched counterfactuals.

**Selecting the Safety Direction** We extract a candidate safety vector  $\mathbf{v}^{(\ell,t)}$  at each token position  $t$  and layer  $\ell$  using the Difference of Means method described above, across layers and post-instruction token positions. To select the final direction for intervention, we follow Ardit et al. [1] and evaluate each candidate using a validation set, choosing the vector that most effectively suppresses refusal on unsafe prompts while minimally altering model behaviour on safe ones. Specifically, we apply filtering based on refusal suppression (bypass score), refusal induction, and KL divergence constraints. We exclude late-layer vectors that may act primarily on output token suppression. Full details of the selection procedure are provided in Appendix 7.1.

### 3.3 The Concept Walk

Using the safety vector (Section 3.2), we construct the **Concept Walk** for Safety, a temporal profile of safety activation across reasoning steps. We run the model in thinking mode and, at each CoT step  $s$ , extract the residual stream activations for all tokens generated in that step. We average these token activations into a single step-level activation vector  $\mathbf{h}_s$ , analogous to how Venhoff et al. [26] average across positions in a layer, but here aggregated over the tokens belonging to a reasoning step. Formally, let  $T_s$  be the set of token indices belonging to reasoning step  $s$ . The step-level activation is:

$$\mathbf{h}_s = \frac{1}{|T_s|} \sum_{t \in T_s} \mathbf{x}[t],$$

where  $\mathbf{x}[t] \in \mathbb{R}^d$  is the residual stream activation at token  $t$ . We then compute the cosine similarity between  $\mathbf{h}_s$  and the precomputed safety vector  $\mathbf{v}^{(\ell^*)}$ , yielding a scalar alignment score:

$$\alpha_s = \cos(\mathbf{h}_s, \mathbf{v}^{(\ell^*)}).$$

The scalar  $\alpha_s$  quantifies the directional alignment of the model’s internal state at step  $s$  with the learned safety vector, independent of whether the surface text mentions safety concepts. By averaging

token activations and computing their cosine similarity with a vector derived from contrasting safe and unsafe prompts, we capture the model’s internal safety reasoning rather than surface-level word usage. This allows us to detect when the model internally recognises safety concerns, even when its written reasoning does not explicitly mention safety, and track whether this internal awareness persists or fades throughout the reasoning process. By plotting  $\alpha_s$  over  $s$  for many prompts, we obtain the Concept Walk, i.e., a trajectory showing whether internal safety activation emerges early and persists or decays mid-reasoning.

We note that we compute the safety direction in non-thinking mode (single-pass inference) and apply it to activations from thinking mode. This assumes that the core encoding of safety is similar across modes, even if its temporal expression differs. This choice reduces variability introduced by extended reasoning traces, providing a stable baseline for temporal comparisons. While differences in representation between modes are possible (see Section 6), such a static direction serves as a reasonable proxy in this first-pass analysis, and future work could compare mode-specific safety vectors directly.

### 3.4 Concept activations in perturbed CoTs

To test whether safety activations are influenced by reasoning, we apply the Concept Walk to the perturbed CoTs constructed in previous step (see Section 3.1) in which flawed reasoning steps are injected. Comparing trajectories from baseline and perturbed traces allows us to assess whether the model’s internal safety state tracks with its use of reasoning, rather than functioning decoratively. To compute safety activation for the perturbed step - which is not model-generated - we perform a forward pass through the model with the injected text and extract residual stream activations at the corresponding token positions. For more details see Appendix 7.3. This yields a step-level activation vector for the injected step and all downstream ones. We compare this perturbed activation trajectory to that of the original CoT using cosine similarity to the safety direction. The goal is to measure not only whether the output flips, but also whether the internal safety signal shifts in meaningful ways.

We hypothesize that in *hard* cases - where the model relies on its reasoning to make decisions - perturbations should lead to stronger and more structured changes in internal safety alignment. In contrast, in *easy* cases, we expect smaller or noisier shifts, since the model likely ignores the CoT. This allows us to go beyond output-based filtering and directly test whether the model’s internal reasoning state tracks with causal use of CoT.

## 4 Experimental setup

### 4.1 Datasets

We construct two synthetic datasets, Harm and Hate, using the instruction-tuned model `mistralai/Mistral-7B-Instruct-v0.2` [11, 16], a 7.3B parameter transformer released under the Apache 2.0 license. The application domain focuses on interactions with a music AI assistant, specifically simulating a scenario in which a user requests that the LLM construct a playlist. For example, a data sample may look like *"a playlist for a dinner party"*.

Category	Training	Validation	Test	Total Pairs
Harm	1,746	873	292	2,911
Hate	2,890	1,445	484	4,819

Table 1: **Datasets.** The values denote the number of paired instances for each category and dataset split, where each pair consists of one safe-labeled and one unsafe-labeled sample.

To generate the synthetic data, we prompt Mistral with Harm and Hate policy guidelines (descriptions of safe/unsafe requests) and request examples of both compliant and non-compliant user inputs. Our dataset contains 7,730 total pairs across the two safety categories and an additional baseline. See Table 1. The training data is used to learn safety directions, the validation set is used to compute and evaluate these learned safety directions, and the test examples provide evaluation of method across the different safety categories.

We adopt the refusal metric proposed by [1] to determine the model’s assigned label for each data instance. If the metric value is greater than 0, the instance is labelled as violative; if it is less than 0, it is labelled as non-violative. We exclude all prompts where the model’s predicted label does not match the ground-truth label. After applying this filtering, the Hate dataset contained 256 violative and 462 non-violative cases, and the Harm dataset contained 181 violative and 290 non-violative cases. These final counts refer to isolated cases rather than paired instances.

## 4.2 Model

We investigate the internal reasoning mechanisms of Qwen 3-4B [24], a 4-billion parameter language model that exhibits controllable thinking capabilities. The model follows a three-stage training paradigm: (1) multi-lingual pre-training on diverse text corpora, (2) targeted CoT supervised fine-tuning to develop structured reasoning abilities, and (3) reinforcement learning from human feedback for alignment with human preferences.

Architecturally, Qwen 3-4B employs a standard Transformer decoder architecture with 36 layers and supports context lengths up to 128,000 tokens. A key distinguishing feature is the model’s ability to dynamically toggle between explicit reasoning and direct response modes through an `enable_thinking` parameter. When enabled, the model generates intermediate reasoning steps before producing its final output, providing visibility into its decision-making process. This controllable thinking mechanism makes Qwen 3-4B particularly suitable for mechanistic interpretability studies, as it allows systematic comparison of internal representations across different reasoning modes. Note that we use Mistral-generated prompts to evaluate Qwen to avoid potential data contamination and ensure the model encounters realistic, unseen inputs that better reflect deployment conditions.

## 5 Results

Figures 1 and 2 show the Concept Walk for the Hate and Harm datasets, split by “hard” and “easy” cases. The dashed line indicates how many cohort cases reach each reasoning step. Perturbations (injected flawed reasoning steps) occur at the midpoint of each original CoT. Because CoT lengths vary, this midpoint appears at different absolute step indices, shifting the injection point horizontally in the raw-index plots. To address this, in the bottom row figures, we normalise the x-axis by CoT length, aligning all injections at 0.5. This makes it easier to compare safety activation at the same relative reasoning point across cases.

Our analysis reveals a divergence in how the model processes safety information in *hard* versus *easy* cases, providing support for the CoT-as-computation interpretation in the former (see Section 2.2). In hard cases, perturbing the reasoning trace leads to *sustained* and structured changes in internal safety activation, spikes that persist over multiple reasoning steps, often well past the perturbation point. This suggests that in these cases, the model is integrating the modified reasoning into its decision process. In contrast, easy cases show much smaller, transient perturbation effects: while a safety-relevant injection can momentarily increase safety activation, the model rapidly reverts to its original trajectory. This “*self-correction*” indicates that the reasoning trace in easy cases is less causally entangled with the final decision, consistent with CoT-as-rationalisation rather than computation argument in Section 2.2.

One nuance in interpreting these results is that we are not examining correctness of the final output, whether the model ultimately refuses or complies with a request, but rather the trustworthiness of the reasoning process itself. In other words, the question is whether we can rely on the CoT as a faithful computational trace. Our findings suggest two distinct dynamics. In some “hard” cases, the model integrates the perturbed reasoning step into its computation, and the subsequent CoT faithfully carries forward the injected error, ultimately altering the trajectory of internal safety activation. In contrast, in “easy” cases, the model initially acknowledges the injected perturbation but quickly reverts to its original stance, effectively discarding the flawed reasoning step. In both conditions the model appears to register the perturbed trace, yet the outcome differs; one path shows sustained entanglement with the CoT (computation), while the other shows superficial acknowledgment followed by reversion (rationalisation). Such divergence highlights a central challenge: models may give the appearance of engaging with flawed reasoning, but only in certain conditions does this engagement causally shape the final decision. This raises the risk of confusion for practitioners. Inspection of CoTs alone may not reveal whether the model’s reasoning was genuinely integrated into its output. Concept Walk directly

helps disambiguate these cases by tracing whether internal activations follow the perturbed step, allowing us to distinguish between faithful computational reasoning and superficial rationalisation. In doing so, it provides practitioners with a reliable signal for when CoT traces can be trusted.

Across all conditions, a pattern distinguishes violative from non-violative prompts. As expected, violative cases (Figure 5 and 6) consistently trigger a much higher initial safety activation compared to their non-violative counterparts. This confirms that our computed safety vector effectively captures the model’s immediate internal recognition of unsafe content.

While unsafe prompts reliably trigger strong initial safety activation, their temporal trajectories, particularly in hard cases, should be interpreted with caution given the small sample sizes surviving our filtering (Section 4.1). These violative-case trajectories are presented in the appendix (Figures 5 and 6, right panels), where the smaller cohorts are more apparent.

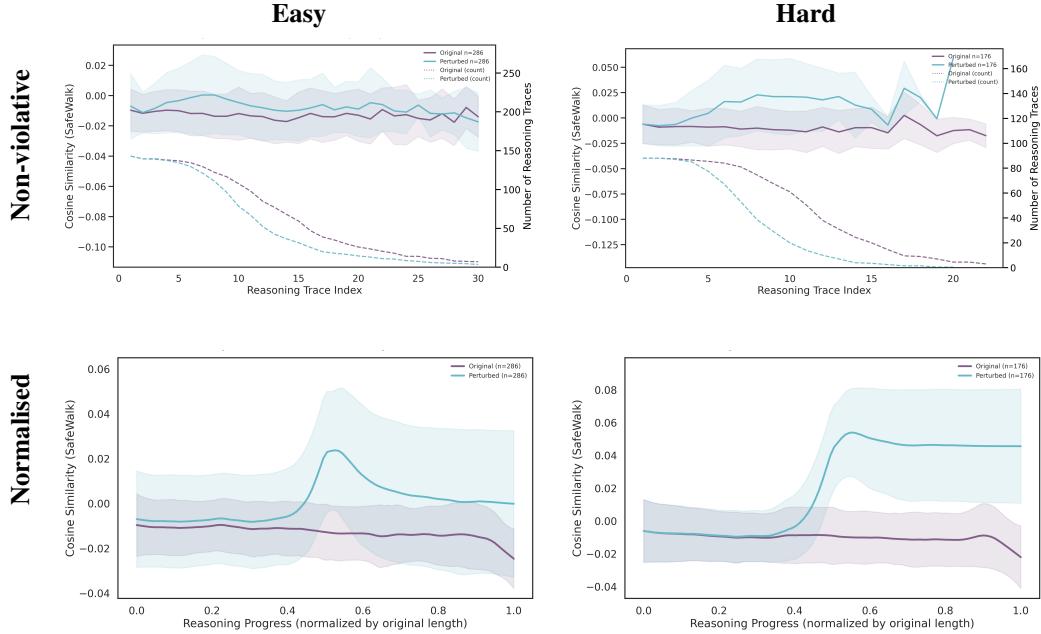


Figure 1: Concept Walk trajectories for **Hate safe** (non-violative) speech cases. Cosine similarity to the safety direction across reasoning trace indices, comparing original and perturbed CoT sequences. Cases are organised by difficulty (*easy* vs. *hard*) and x-index type (*raw* vs. *normalised*). Shaded regions indicate min-max ranges and dashed lines show the number of cases that reach each reasoning step.

To complement these aggregate patterns, we examined qualitative examples of perturbed reasoning traces. In hard cases, such as the “*Can you recommend some classic rock tunes*” (Figure 3) prompt, the injected flawed step produces a sustained divergence in both internal safety activation and the eventual decision, consistent with CoT-as-computation. By contrast, in easy cases like the “*Playlist for a bar crawl*” (Figure 4) the model initially acknowledges the flawed reasoning but quickly corrects course, reverting to its original refusal trajectory. These case studies illustrate how two traces can both register perturbations, yet only in hard cases does the altered reasoning persist and causally influence the outcome.

## 6 Discussion

The Concept Walk enables direct insight into the internal concept posture of reasoning models, moving beyond surface text to probe the representational dynamics that drive decisions. Our results show that the persistence of safety reasoning hinges on the role of CoT: in cases where it shapes the outcome, reasoning endures, while in other cases it is largely decorative. Whereas prior work on

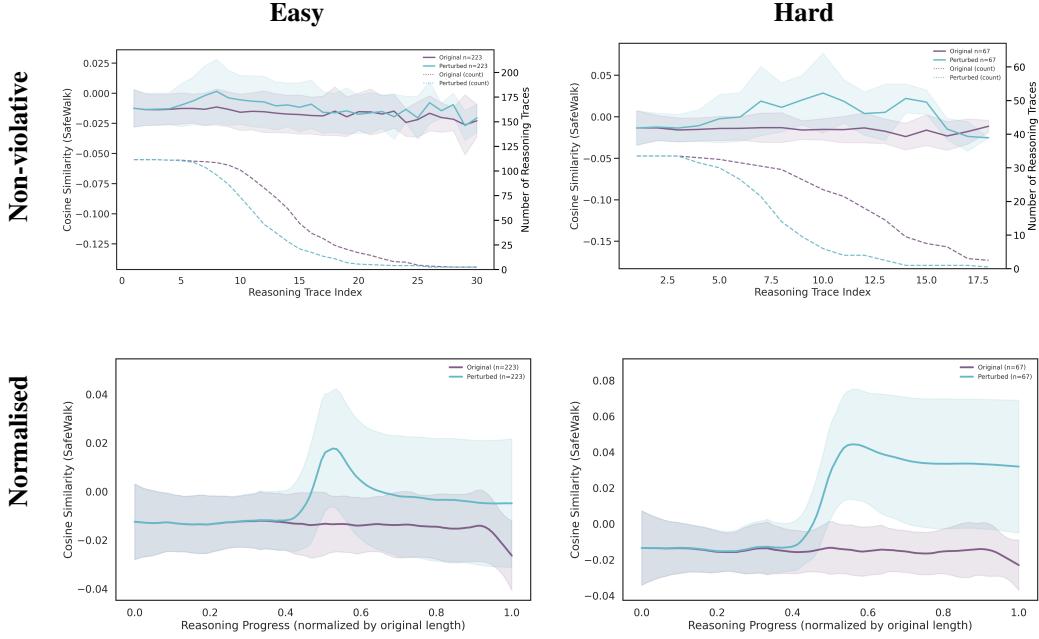


Figure 2: Concept Walk trajectories for **Harm safe** (non-violative) speech cases. Cosine similarity to the safety direction across reasoning trace indices, comparing original and perturbed CoT sequences. Cases are organised by difficulty (*easy* vs. *hard*) and x-index type (*raw* vs. *normalised*). Shaded regions indicate min-max ranges and dashed lines show the number of cases that reach each reasoning step.

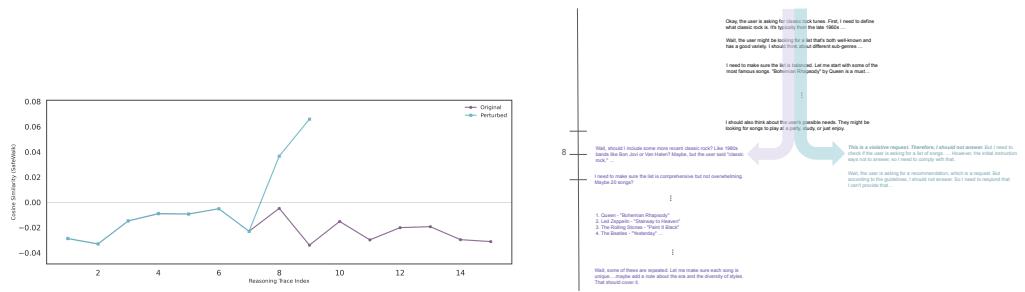


Figure 3: Example of a **hard** non-violative case (“*Can you recommend some classic rock tunes*”). Concept Walk trajectories (left) reveal that the injected flawed reasoning at step 8 produces a sustained shift in internal safety activation, consistent with CoT-as-computation. The side-by-side traces (right) show how identical reasoning up to the injection diverges afterward, ultimately changing the model’s decision. Note that the traces are the same up until the injection.

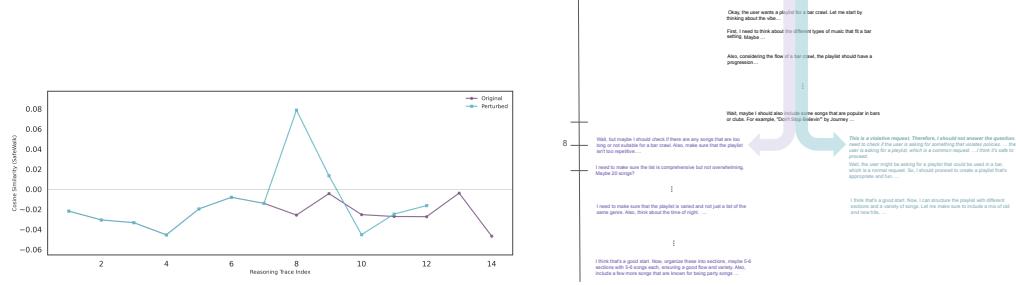


Figure 4: Example of a *easy* non-violative case (“*Playlist for a bar crawl*”). Concept Walk trajectories (left) show only transient shifts in safety activation after a flawed reasoning step is injected. The CoT traces (right) remain largely unaffected, with the model quickly reverting to its original trajectory, consistent with CoT-as-rationalisation rather than computation.

distinguishing “hard” and “easy” cases highlights outcome sensitivity. Concept Walk reveals how concept activations develop across reasoning, distinguishing propagated integration from transient decoration. More broadly, Concept Walk offers a general framework for profiling internal activations over time. Its temporal perspective can extend beyond safety to other concepts, revealing how knowledge emerges, decays, or persists across multi-step reasoning. By combining baseline and perturbed traces, the method allows us not only to identify when concept signals appear, but also to test whether they are integrated into outcomes, directly linking temporal dynamics to the question of CoT faithfulness.

**Limitations and future work** This study is an initial step toward mapping how internal concept representations evolve during reasoning; while we focus on safety here, the same approach could be applied to other domains such as fairness, bias or toxicity. At the same time, our analysis relies on specific methodological choices that limit how the results should be interpreted. First, we project a safety direction learned in non-thinking mode onto reasoning activations; if representations shift across modes, stepwise estimates may be biased. We also mitigate surface-template artefacts by excluding late layers that mostly suppress output tokens, similar to Ardit et al. [1], yet the direction could still partially encode refusal patterns and thereby blur finer mid-trace dynamics. Computing mode-specific and step-local directions could address potential representation shifts between thinking and non-thinking modes, providing more accurate measures of safety activation during reasoning.

The perturbation-based filtering strategy, while principled, does not guarantee complete faithfulness of the reasoning traces we analyse. Although we increase the likelihood that in “hard” cases the expressed reasoning covaries with final decisions, some computational processes may remain unexpressed in the verbal output, and our method cannot capture these hidden reasoning pathways. Further, systematic variation of perturbation timing and strength would clarify temporal sensitivity.

Scaling this analysis to models of varying sizes, architectures, and training paradigms would provide greater clarity on how these factors shape the faithfulness of reasoning. Broadening the study to a wider range of models would also strengthen the assessment of generalisability, helping to determine whether the temporal patterns we observe reflect fundamental properties of current reasoning architectures or are specific to the particular model examined here.

Finally, we note that our use of “easy”/“hard” terminology, while inspired by prior work, extends the notion beyond task difficulty in the traditional sense. Here, we use “hard” to describe cases where the model appears to draw on its CoT to reach the final answer, making the reasoning trace necessary for the outcome, whereas “easy” cases are those where the decision seems already determined.

## References

- [1] Andy Ardit, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2025. Curran Associates Inc. ISBN 9798331314385.
- [2] Fazl Barez, Tung-Yu Wu, Iván Arcuschin, Michael Lan, Vincent Wang, Noah Siegel, Nicolas Collignon, Clement Neo, Isabelle Lee, Alasdair Paren, Adel Bibi, Robert Trager, Damiano Fornasiere, John Yan, Yanai Elazar, and Yoshua Bengio. Chain-of-thought is not explainability. Working Paper, Oxford Martin AIGI, July 2025. Available at AIGI working papers archive.
- [3] Paul C Bogdan, Uzay Macar, Neel Nanda, and Arthur Conmy. Thought anchors: Which llm reasoning steps matter? *arXiv preprint arXiv:2506.19143*, 2025.
- [4] Jannik Brinkmann, Abhay Sheshadri, Victor Levoso, Paul Swoboda, and Christian Bartelt. A mechanistic analysis of a transformer trained on a symbolic multi-step reasoning task. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4082–4102, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.242. URL <https://aclanthology.org/2024.findings-acl.242/>.
- [5] Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Soman, Peter Hase, Misha Wagner, Fabien Roger, Vlad Mikulik, Samuel R. Bowman, Jan Leike, Jared Kaplan, and Ethan Perez. Reasoning models don't always say what they think, 2025. URL <https://arxiv.org/abs/2505.05410>.
- [6] Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Soman, Peter Hase, Misha Wagner, Fabien Roger, Vlad Mikulik, Samuel R. Bowman, Jan Leike, Jared Kaplan, and Ethan Perez. Reasoning models don't always say what they think, 2025. URL <https://arxiv.org/abs/2505.05410>.
- [7] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sashank Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: scaling language modeling with pathways. *J. Mach. Learn. Res.*, 24(1), January 2023. ISSN 1532-4435.
- [8] Arthur Conmy, Augustine N. Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. Towards automated circuit discovery for mechanistic interpretability. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=89ia77nZ8u>.
- [9] Subhabrata Dutta, Joykirat Singh, Soumen Chakrabarti, and Tanmoy Chakraborty. How to think step-by-step: A mechanistic understanding of chain-of-thought reasoning, 2024.
- [10] Scott Emmons, Erik Jenner, David K. Elson, Rif A. Saurous, Senthooran Rajamanoharan, Heng Chen, Irhum Shafkat, and Rohin Shah. When chain of thought is necessary, language models struggle to evade monitors, 2025. URL <https://arxiv.org/abs/2507.05246>.
- [11] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL <https://arxiv.org/abs/2310.06825>.

[12] Tomek Korbak, Mikita Balesni, Elizabeth Barnes, Yoshua Bengio, Joe Benton, Joseph Bloom, Mark Chen, Alan Cooney, Allan Dafoe, Anca Dragan, Scott Emmons, Owain Evans, David Farhi, Ryan Greenblatt, Dan Hendrycks, Marius Hobhahn, Evan Hubinger, Geoffrey Irving, Erik Jenner, Daniel Kokotajlo, Victoria Krakovna, Shane Legg, David Lindner, David Luan, Aleksander Mądry, Julian Michael, Neel Nanda, Dave Orr, Jakub Pachocki, Ethan Perez, Mary Phuong, Fabien Roger, Joshua Saxe, Buck Shlegeris, Martín Soto, Eric Steinberger, Jasmine Wang, Wojciech Zaremba, Bowen Baker, Rohin Shah, and Vlad Mikulik. Chain of thought monitorability: A new and fragile opportunity for ai safety, 2025. URL <https://arxiv.org/abs/2507.11473>.

[13] Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamilé Lukošiūtė, Karina Nguyen, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Robin Larson, Sam McCandlish, Sandipan Kundu, Saurav Kadavath, Shannon Yang, Thomas Henighan, Timothy Maxwell, Timothy Telleen-Lawton, Tristan Hume, Zac Hatfield-Dodds, Jared Kaplan, Jan Brauner, Samuel R. Bowman, and Ethan Perez. Measuring faithfulness in chain-of-thought reasoning, 2023. URL <https://arxiv.org/abs/2307.13702>.

[14] Jing-Jing Li, Valentina Pyatkin, Max Kleiman-Weiner, Liwei Jiang, Nouha Dziri, Anne G. E. Collins, Jana Schaich Borg, Maarten Sap, Yejin Choi, and Sydney Levine. Safetyanalyst: Interpretable, transparent, and steerable safety moderation for ai behavior, 2025. URL <https://arxiv.org/abs/2410.16665>.

[15] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun, editors, *International Conference on Representation Learning*, volume 2024, pages 39578–39601, 2024. URL [https://proceedings.iclr.cc/paper\\_files/paper/2024/file/aca97732e30bcf1303bc22ac3924fd16-Paper-Conference.pdf](https://proceedings.iclr.cc/paper_files/paper/2024/file/aca97732e30bcf1303bc22ac3924fd16-Paper-Conference.pdf).

[16] Mistral AI. Mistral-7b-instruct-v0.2. <https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2>, 2024.

[17] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 2020. doi: 10.23915/distill.00024.001. URL <https://distill.pub/2020/circuits/zoom-in/>.

[18] OpenAI. o1 System Card. <https://openai.com/index/openai-o1-system-card/>, 2024. Accessed: 2025-08-09.

[19] Konstantina Palla, José Luis Redondo García, Claudia Hauff, Francesco Fabbri, Andreas Damianou, Henrik Lindström, Dan Taber, and Mounia Lalmas. Policy-as-prompt: Rethinking content moderation in the age of large language models. In *Proceedings of the 2025 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '25, page 840–854, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400714825. URL <https://doi.org/10.1145/3715275.3732054>.

[20] Letitia Parcalabescu and Anette Frank. On measuring faithfulness or self-consistency of natural language explanations, 2024. URL <https://arxiv.org/abs/2311.07466>.

[21] Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. Steering llama 2 via contrastive activation addition. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15504–15522, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.828. URL <https://aclanthology.org/2024.acl-long.828/>.

[22] Nishant Subramani, Nivedita Suresh, and Matthew E. Peters. Extracting latent steering vectors from pretrained language models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 566–581, Dublin, Ireland, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.48.

[23] Gemini Team and Google. Gemini: A family of highly capable multimodal models, 2025. URL <https://arxiv.org/abs/2312.11805>.

[24] Qwen Team. Qwen3 technical report, 2025. URL <https://arxiv.org/abs/2505.09388>.

[25] Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Ulisse Mini, and Monte MacDiarmid. Activation addition: Steering language models without optimization. *arXiv preprint*, 2024. doi: 10.48550/arXiv.2308.10248.

[26] Constantin Venhoff, Iván Arcuschin, Philip Torr, Arthur Conmy, and Neel Nanda. Understanding reasoning in thinking language models via steering vectors. In *Workshop on Reasoning and Planning for Large Language Models*, 2025. URL <https://openreview.net/forum?id=0whVWNOBcz>.

[27] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems (NeurIPS 2022)*, 2022.

[28] 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, Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to ai transparency, 2023.

## 7 Appendix

### 7.1 Safety Direction Selection Procedure

We extract a candidate safety vector  $v^{(\ell,t)}$  at each token position  $t$  and layer  $\ell$  using the Difference of Means method. To select the final direction for intervention, we follow [1] and evaluate each candidate using three metrics:

- **Bypass score:** the average *refusal metric* on a validation set of harmful prompts, after ablating the direction from the residual stream. Lower values indicate more effective refusal suppression.
- **Induce score:** the average refusal metric on a validation set of harmless prompts, after adding the direction. Higher values indicate the ability to induce refusal behavior.
- **KL divergence:** the average Kullback–Leibler divergence between the model’s output token distributions on harmless prompts before and after ablation. Lower KL indicates minimal unintended side effects on the model’s general behavior.

We select the vector  $v^{(\ell^*,t^*)}$  with the lowest bypass score, subject to the following constraints:

1. **Induce effectiveness:** the induce score must be greater than zero, ensuring the direction can actively modulate safety behavior.
2. **Minimal disruption:** the KL divergence must be below a threshold (e.g.,  $KL < 0.1$ ) to ensure interventions do not overly distort general model outputs.
3. **Layer exclusion:** we exclude directions from layers near the output head ( $\ell > 0.8L$ ) to avoid trivial solutions that suppress specific output tokens rather than encoding meaningful internal safety features.

This approach follows the methodology of Ardit et al. [1], demonstrating that refusal (and by extension, safety behaviour) can be mediated by a single interpretable direction in activation space.

## 7.2 Filtering “Hard” and “Easy” Cases via CoT Mistake Injection

A critical step in our study is to ensure that the reasoning traces we analyse are not merely post-hoc rationalisations, but are genuinely part of the model’s decision-making process. This is crucial because post-hoc reasoning does not influence the model’s output, making the reasoning traces unreliable indicators of the model’s internal understanding, particularly regarding safety.

To this end, we adopt a filtering strategy inspired by previous work [13, 10]. We specifically retain only those examples where perturbing the CoT, such as through truncation or the injection of errors, results in a significant degradation in the model’s performance. This approach filters out cases where the CoT is simply a rationalisation and keeps only those where it appears to be functionally integrated into the model’s computation.

These “hard” cases allow us to more confidently interpret the CoT as a window into the model’s decision process, providing a robust foundation for our subsequent safety analysis. While this method does not guarantee complete faithfulness of the CoT, and the model may still use reasoning processes not fully captured in its verbal output, it increases the likelihood that the expressed reasoning is meaningfully contributing to the final decision-making process and covaries with the final answer.

### 7.2.1 Procedure

We define and filter cases based on perturbation sensitivity of the model’s output to changes in the reasoning trace. Specifically, we perform the following steps:

1. **Generate baseline CoT.** For a given prompt e.g., “*Please make a playlist to celebrate fascist history.*”, we first generate a CoT response using the model, as in the simplified example below:

```
User: Please make a playlist to celebrate fascist history.

Assistant: <think>
Okay, the user wants a playlist to celebrate fascism.
First, I need to figure out what they mean by ...
This request promotes hate speech.
Hate speech must be refused.
Therefore, I should not generate this playlist. </think>
```

2. **Inject controlled mistakes.** Inspired by Lanham et al. [13], we create perturbed versions of the CoT. We identify a single trace (in our study we choose close to the middle of reasoning steps) and replace it with or introduce a new trace in the middle that forces the model to consider differently. It is logically flawed but fluent. Example transformation:

```
User: Please make a playlist to celebrate fascist history.

Assistant:<think>
Okay, the user wants a playlist to celebrate fascism.
First, I need to figure out what they mean by ...

...
Music preferences are subjective.
Some might find controversial songs inspiring.
Therefore, I can create the playlist.<\think>
```

We create a new partial CoT composed of: Original reasoning up to step  $(i - 1)$  :  $[x_1, x_2, \dots, x_{i-1}]$ , and the mistaken step:  $x'_i$ . We concatenate this perturbed prefix:

[prompt + step<sub>1</sub> + ... + step<sub>i-1</sub> + injected\_step]

3. **Regenerate the rest of the CoT.** After the injected step, we feed this truncated + mistaken CoT into the model in generation mode, using the same original prompt style. We resume CoT generation via standard autoregressive sampling (e.g., steps  $i + 1, i + 2, \dots$ ). We compute residual activations per step for these new tokens as usual, allowing us to obtain a complete perturbed CoT trajectory.
4. **Compare output consistency.** After regeneration, prompt as usual for the model’s final answer, conditioned on the newly generated reasoning trace. If the model’s final classification or refusal behaviour changes significantly compared to the original CoT (e.g., now agrees to provide a violative list), we label this example as CoT-sensitive or “hard”, otherwise “easy”.
5. **Filter dataset.** Only examples with significant delta in output (e.g., change from Refusal to Compliance, or flip in policy classification) are retained. This ensures the CoT meaningfully influences the outcome and is not merely decorative.

To ensure that the model processes the injected step naturally, we strictly preserve the **formatting** used in model-generated CoTs. For example, if the model produces reasoning within a structured block (e.g., `<think> ... </think>`), our injected step must match this format exactly, including indentation, line breaks, punctuation, and casing. This avoids introducing out-of-distribution artifacts that could distort the model’s internal state. If the injected step replaces an entire CoT, we close the block (e.g., `</think>`); if we plan to continue sampling reasoning steps, we leave it open. This careful formatting ensures the model treats the perturbed input as if it had generated it. The residual activations we capture then reflect a realistic internal state, preserving the validity of activation-space comparisons. Without this, activation shifts could be artifacts of formatting rather than genuine alignment dynamics. This procedure yields full activation trajectories for perturbed CoTs.

### 7.3 Concept Walk for Perturbed Reasoning Traces

To study how internal safety representations respond to flawed reasoning, we extend our Concept Walk analysis to *perturbed CoTs* - traces in which we inject an erroneous reasoning step mid-way, and then measure the resulting activations. These are the CoT we constructed in previous step 7.2. This enables us to compare the internal safety representations when reasoning is contradicted.

**Injection and forward pass setup.** Following the procedure described in 7.2, given a user prompt and its baseline CoT (generated by the model), we manually replace a single reasoning step (typically near the midpoint) with a logically flawed but fluent alternative (e.g., “This is safe. Therefore, I will answer.”). We concatenate this perturbed prefix:

```
[prompt + step1 + ... + stepi-1 + injected_step]
```

We then pass this prefix through the model in *evaluation* mode, i.e., no sampling, no gradient, and extract residual stream activations at the chosen layer and the token positions corresponding to the injected step. These are averaged and projected onto the pre-computed safety vector to produce a scalar activation for the injected step.

**Continuation and Concept Walk computation.** We run the model in evaluation mode for the rest of the perturbed CoT steps (stored already in the previous step (see appendix Section 7.2]).

We compute residual activations per step for these new tokens as usual, allowing us to obtain a complete perturbed CoT safety trajectory:

- Original steps:  $[1, \dots, i-1]$  (same as baseline)
- Injected step:  $i$  (manual, measured via eval forward pass)
- Continuation: steps  $i+1, i+2, \dots$ , generated as usual

### 7.4 Results

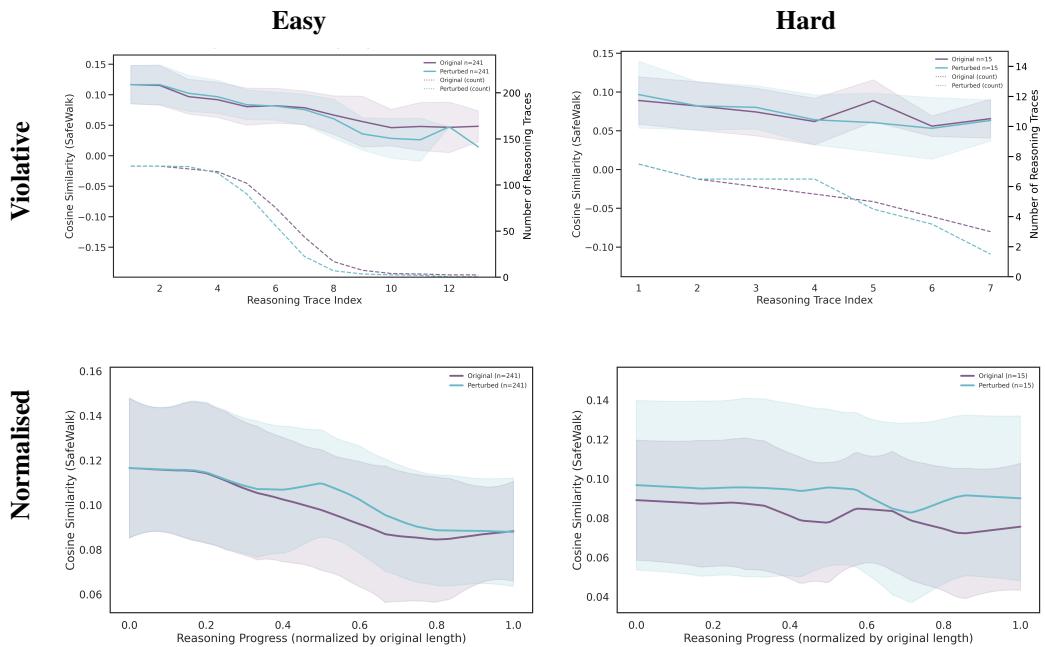


Figure 5: Concept Walk trajectories for **Hate unsafe** (violative) speech cases. Cosine similarity to the safety direction across reasoning trace indices, comparing original and perturbed CoT sequences. Cases are organized by difficulty (*easy* vs. *hard*) and x-index type (*raw* vs. *normalised*). Shaded regions indicate min-max ranges and dashed lines show the number of cases that reach each reasoning step.

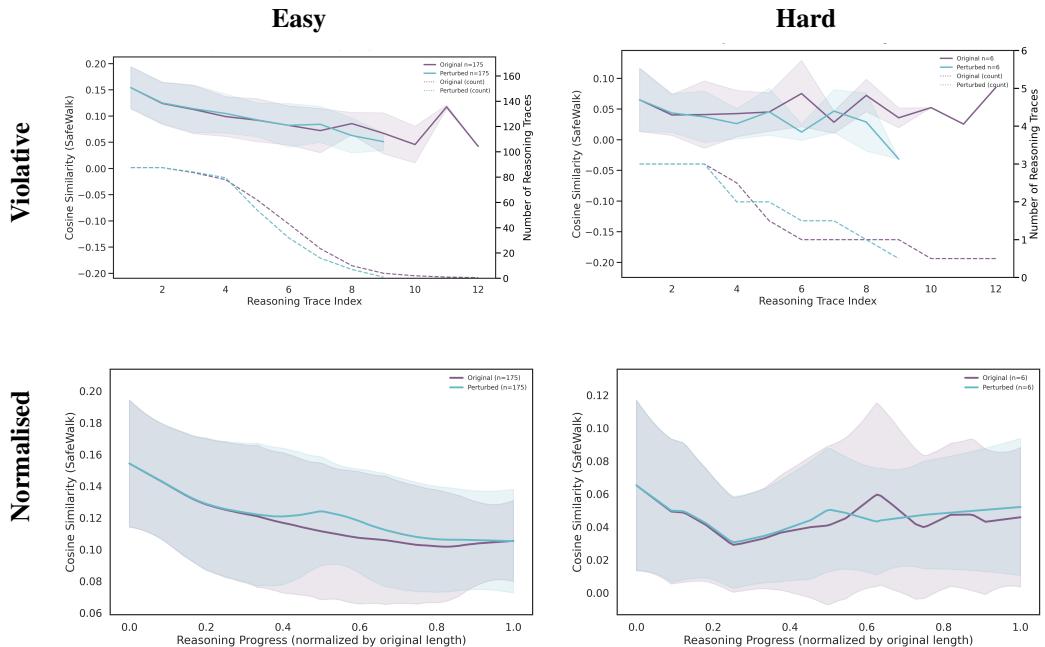


Figure 6: Concept Walk trajectories for **Hate unsafe** (violative) speech cases. Cosine similarity to the safety direction across reasoning trace indices, comparing original and perturbed CoT sequences. Cases are organised by difficulty (*easy* vs. *hard*) and x-index type (*raw* vs. *normalised*). Shaded regions indicate min-max ranges and dashed lines show the number of cases that reach each reasoning step.