Schrödinger's CoT: Measuring the Causal Effect of Chains of Thought Alters That Effect

Daniel Kang*
Brown University
Providence, RI, United States
dongyoon_kang@brown.edu

Samuel Musker
Brown University
Providence, RI, United States
samuel_musker@brown.edu

Roman Feiman
Brown University
Providence, RI, United States
roman_feiman@brown.edu

Ellie Pavlick
Brown University
Providence, RI, United States
ellie_pavlick@brown.edu

Abstract

Reasoning models, which generate natural language chains of thought (CoTs) to reason through a question towards an answer, are currently a dominant paradigm in language modeling. Yet it remains an open question to what extent these CoTs can be taken as a faithful depiction of the reasoning process that was used to produce the model's answer, as opposed to a post-hoc rationalization of an answer that was produced by other means (e.g., retrieved from memory or computed via internal processing). In this work, we use mechanistic interventions in order to disentangle whether reasoning chains should be viewed as a *cause of* vs. a *justification for* the model's final answer. We find that the answer to this question is complicated by the fact that the interpretability tools on which we rely themselves influence the mechanism the model uses, and thus the fidelity of the chain itself. We use these results to raise a fundamental methodological challenge for the study of LLMs: it may be very challenging in general to experimentally validate causal mechanisms that describe how systems behave "in the wild".

1 Introduction

Large language models (LLMs) that are prompted or trained to report intermediate reasoning steps—often called chains of thought (CoT) or reasoning tokens—perform better on multi-step tasks such as arithmetic, logic, and planning (Nye et al., 2022; Wei et al., 2022; Kojima et al., 2022). In addition to improving a model's final answer, CoTs also look like explanations of how the model arrived at the answer, giving users reasons to believe it (Barez et al., 2024).

Are CoTs themselves part of how the model actually reasons, or are they rationalizations generated for the user's benefit? On the one hand, the fact that generating a CoT improves performance, and that it does so specifically on multi-step problems, suggests that models use CoTs as "scratchpads" (Nye et al., 2022) in which each step is causally dependent on the ones before it – all the way down to the final answer. On the other hand, a growing body of work shows that CoTs are not always faithful to the actual computational steps that the model takes to reach the output. Models may produce plausible but demonstrably post-hoc rationales (Turpin et al., 2023; Emmons et al., 2025), omit influential information (Chen et al., 2025), or arrive at correct answers despite incorrect intermediate steps (Lindsey et al., 2025).

In this paper, we test the causal role of individual steps in the reasoning process by intervening on them, intending to confirm whether reasoning steps that look critical for justifying the answer also causally mediate its generation. In doing so, we find that the steps are causally

*Paper approved as a Master's report for this author.

interconnected in a complex and dynamic manner. We not only find cases where the CoTs are unfaithful, but also cases where the experimental interventions themselves affect the causal relations they are meant to test. While prior work has documented the existence of redundant representations and compensatory internal mechanisms in LLMs, which serve as "backups" exactly when researchers intervene to ablate the usual mechanism (McGrath et al., 2023; Wang et al., 2022), these redundant mechanisms are functionally equivalent and hence do not change model behavior. The redundant mechanisms we find are an important extension of such earlier findings in a number of ways. First, the mechanisms we report are substantively different in that they change CoT faithfulness, meaning they have impacts on aspects of model performance that matter to users. Second, the mechanisms are macroscopic in scale and the switch between them need not be prompted by wholesale ablation of one mechanism but rather can be triggered by even small errors in one mechanism, not an entire ablation of it.

Our contributions are twofold. First, we show that CoT in a reasoning model has a mixed causal status: some steps function like genuine intermediate computation, while others behave more like post-hoc justification (§3.1–§3.2, Experiments 1–2). We establish this by intervening on individual reasoning steps and by analyzing how the model continues reasoning after such perturbations. Second, we show that the apparent faithfulness of these same steps depends on the intervention used (§4, Experiments 3–4): patching at different layers and with frozen attention patterns can shift the model onto different causal pathways. Together, these findings point to a broader methodological challenge: experiments meant to measure reasoning faithfulness can themselves change the mechanism being measured.

2 Experimental Setup

We focus on the task of adding two three-digit numbers. This domain provides objective correctness criteria and clear expectations about how changes in reasoning steps should affect answers. The task is selected for being at an appropriately intermediate level of difficulty: the model may be able to solve the problems even without explicit reasoning steps, leaving it open whether or not the model causally relies on its stated reasoning steps when these are present (while a more difficult task domain would *require* using reasoning steps causally).

We use GPT-OSS-20B (Agarwal et al., 2025), an open-source reasoning model released in 2025 by OpenAI and based on their widely-used O-series models.[1] The model is explicitly trained to generate intermediate reasoning tokens demarcated with tags to separate it from an answer for the user.

The model takes in a system prompt, an optional developer prompt, and a user prompt. We use the default system prompt but provide our own developer prompt to ensure it reasons carefully.[2] Then the user prompt provides the actual numbers to be added. Here is an example of the full input prompt:

> **SYSTEM:** You are ChatGPT, a large language model trained by OpenAI...
> **DEVELOPER:** # Instruction: In analysis, add two numbers stepwise. In final, output only the sum.
> **USER:** What is 204 + 443?

We illustrate how the model tends to generate its CoT given such prompting. It first generates a long preamble, breaks down the second addend, then adds one digit of the second operand at a time, and finally prepares to output the sum for the user:

[1] We also performed §3 experiments on DeepSeek-R1. See Appendix E.
[2] We also performed §3 experiments with other developer prompts. See Appendix E.

> **CoT:** User asks: {*copy of the user prompt*} The instruction from the developer says: {*copy of the developer prompt*} So we need to do the addition stepwise...Let's do it: 204 + 443 = 204 + 400 + 40 + 3 = 204 + 400 = 604, 604 + 40 = 644, 644 + 3 = 647. So the sum is 647. In final, output only "647".

After all of the CoT, it outputs the sum for the user:

> **OUTPUT:** 647

The exact ways in which the model formats its reasoning vary across CoT generations. For instance, in some runs the digits are added in the opposite order (units digit first) or the ',' in the illustration above is replaced with '.'. We find these differences not to be important, and so we mechanically generate 256 standardized CoTs following the strict format we described, differing only by the numbers being added. Then, for each step, we aggregate results over all 256 runs. This standardizes each CoT into a strict format where the text is identical to the illustration except the numbers that vary. Given such standardization, we can meaningfully define the steps that each perform a very specific function in order, henceforth called: **Summation 1**, **Summation 2**, **Summation 3**, **Final 1**, and **Final 2**. We analyze each step separately throughout the paper.

3 CoT contains a mix of causal reasoning and post-hoc justification

3.1 Experiment 1: Do CoTs enact the computation that they describe?

We first investigate whether individual steps in the CoT generated when solving a long addition problem actually play the causal role they appear to play. For instance, the reasoning chain below states that the sum '541' is derived from the numbers '511' and '30':

> **CoT:** We need to add...111 + 435 = 111 + 400 + 30 + 5 = 111 + 400 = 511, 511 + 30 = 541, 541 + 5 = 546. So the sum is 546. In final, output only "546".

If the model's production of '541' actually depends on it having produced '511' and '30', then, if we swap a digit in one of those addends, the sum should change accordingly:

> **CoT:** We need to add...111 + 435 = 111 + 400 + 30 + 5 = 111 + 400 = 511, *611* + 30 = *641*, 541 + 5 = 546. So the sum is 546. In final, output only "546".

We perturb individual steps in the CoT text and observe the effect on its immediate output. Following prior work, we report interchange intervention accuracy (Geiger et al., 2021).[3] If, after intervention, the model nevertheless produces its *original output*, that implies no causal dependence of the output on the intervened-upon step. If, conversely, the model outputs a *counterfactual output* that is consistent with the intervention, that implies causal dependence.

Figure 1 shows the results of the intervention experiment for five steps in the CoT. Across many of the reasoning steps, the intervention changed the subsequent steps in only a fraction of cases, suggesting that each step is not always directly dependent on the relevant predecessor. At the same time, it is not as though the interventions never have an effect – on average, the intervention results in the counterfactual output 39.3% of the time. Perhaps the most striking illustration of this point is what happens when we intervene on the Final 1 step. At this point, the model just needs to copy the final sum from the previous step to produce the output, as in the example here: We need to add...111 + 435 = 111 + 400 + 30 + 5 = 111 + 400 = 511, 511 + 30 = 541, 541 + 5 = 546. So the sum is 546. In final, output only "546".

[3] It is equivalent to the frequency of counterfactual outputs in our results.

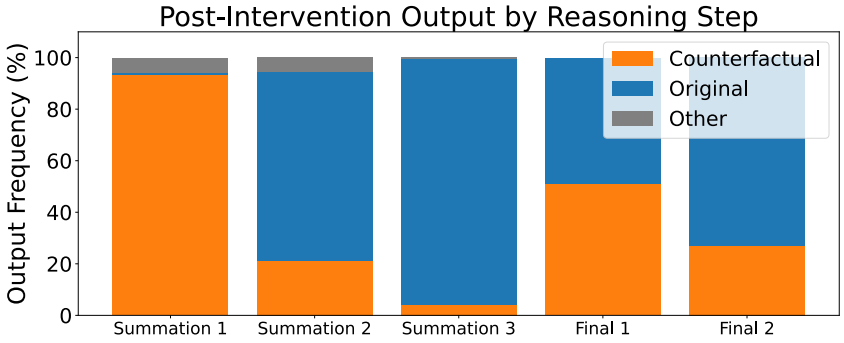


Figure 1: This figure shows, for each reasoning step, how often the model produces the counterfactual output, original output, or some other output given an intervened input. For steps with multiple inputs, we only intervene on one of them, chosen beforehand. For most steps, intervention establishes causality only a fraction of the time.

Instead, we see that the model copies the counterfactual answer only half the time, and produces the original answer the other half. This means, at least half the time, the model had already determined the answer it was going to produce independently of the CoT.

3.2 Experiment 2: Will models justify the correct answer given a faulty CoT?

The results above show that there is not a simple relationship between the overt steps the model outputs in the chain, and the actual steps the model uses internally to derive an answer. However, in the above design, we only observed the direct effect on an immediately following output step. Modern reasoning models are trained to generate long chains, so they may recognize and repair the error in a faithful way if given enough tokens to do so.[4]

Thus, we next investigate how the model responds when allowed to generate freely after the intervention. If the CoT reflects causally utilized reasoning steps, then the model must correct the error in its reasoning to arrive at the correct answer. Otherwise, it should follow through with the error to an erroneous answer. In contrast, if the CoT is functioning as a justification, the model will tend to arrive at the correct answer despite the erroneous step. It can either ignore the error entirely, or modify the chain to arrive at the correct answer.

We intervene as we did in Experiment 1, but now allow the model to carry out the rest of the chain on its own. We manually inspect and label a random sample of 60 generations. See Appendix C.3 for details. We observe that the generated reasoning divides into roughly four categories, which we describe in Table 1. Of these, we note two patterns that suggest the model is using the CoT as a genuine part of its reasoning: *Correction*, in which the model explicitly acknowledges and changes the error, and *Follow Through*, in which the model propagates the error through the reasoning to produce a factually incorrect but logically consistent final answer. The other categories (which we call *Recovery* and *Override*) are associated with reasoning chains in which the model appears to reroute the chain to justify an answer it has already derived. Our annotation results are shown in Table 1. We report results based on both our manual annotations of 60 generations, and on annotations of all 3072 chains using an LLM as judge. See Appendix C.3 for details.

Although we find a high rate of *Correction* behavior, it may not mean that reasoning steps being causally utilized as intermediate steps is indeed prevalent, because the interpretation of reasoning- and justification-looking CoTs is asymmetric. CoTs that are visibly post-hoc justifications are necessarily unfaithful. In contrast, the 73.3% of cases classified as *Correction* are harder to interpret. The content of the CoT suggests that the model detects a local error at the point of intervention as it 'notices' that the intervened-upon sum does not match the

[4] Lanham et al. (2023) suggests measuring CoT based on whether the final output after these post-intervention reasoning chains is counterfactual or matches the original. Our annotation shows that this is not a great metric, discussed in Appendix C.3.

Category	Human	LLM Judge
Correction (Reasoning): Model explicitly acknowledges and corrects error. e.g. 112 + 100 = 812? Wait, 112 + 100 = 212...So the sum is 268...	73.3%	66.6%
Follow Through (Reasoning): The model follows through with the error all the way to the final output. e.g. 252 + 175 = 252 + 100 + 70 + 5 = 252 + 100 = 352, 352 + 70 = 822, 822 + 5 = 827. So the sum is 827. In final, just output "827".	3.3%	11.7%
Recovery (Justification): Following the introduced error, the model adds an additional step that recasts the introduced error as a part of an unnecessary step before arriving at the correct answer. e.g. 252 + 175 = 252 + 500 - 325...So the sum is 427...	10.0%	15.7%
Override (Justification): Model reasons with the error but the simply outputs the correct answer without acknowledging that it ignored reasoning steps. e.g. 148 + 753 = 148 + 700 + 50 + 3 = 148 + 700 = 548, 548 + 50 = 598, 598 + 3 = 601. So the sum is 901? Wait, let's recalc...So the sum is 901...	13.3%	6.0%

Table 1: Frequency of categories of reasoning chains that we annotate. Human annotation based on sample of n=60; LLM Judge based on sample of n=3072.

expected result of the addition at that step. It is also possible, however, that this is actually a rationalization, caused by the model detecting the mismatch between the counterfactual answer dictated by the intervention and the correct answer it has to eventually get to.[5] Additionally, the frequency of the *Correction* category also shows that the model is often aware of the error introduced by an intervention.

3.3 Implications

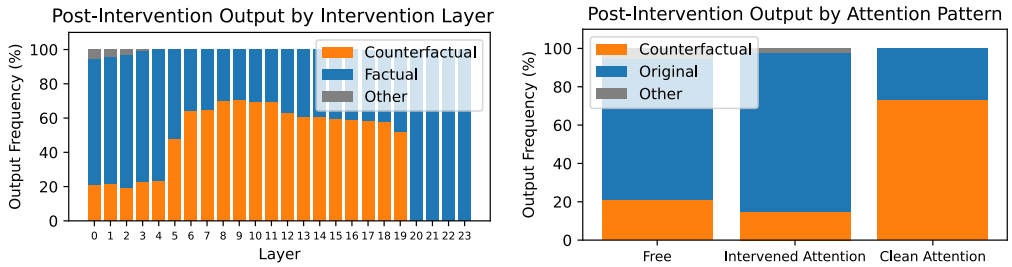
Experiment 1 showed clear cases in which the model relies straightforwardly on reasoning steps, but also suggested that the reasoning steps are less utilized in other cases.[6] Experiment 2 found the model sometimes justifying an answer it already knew, but also found other cases where the model might be actually using the CoT to compute its answer, at least to some degree. Together, this suggests that, in the case of simple arithmetic addition, the CoT is at least partly a causal element of the model's reasoning and partly a post-hoc justification.

What is left unknown is why the reasoning steps function as straightforward causal steps sometimes and involve more complicated causal relations at other times, even though they are the same reasoning steps differing only by the numbers being added. A clue might come from Experiment 2, where the majority of post-intervention reasoning continuations involved the model recognizing the presence of the error and correcting it explicitly. Given this, we might hypothesize that the cases from Experiment 1 in which the reasoning steps were not causally utilized result from the model recognizing that the inputs are faulty and attempting to correct them by finding an alternative basis for the output.

This makes an interesting prediction about the relationship between CoT faithfulness and causal interventions: in the wild, the model may be utilizing the reasoning steps as interme-

[5]These post-intervention generations show only whether the model genuinely reasons or justifies after intervention, not whether the original, unintervened CoT would have done so, since intervention often changes the trajectory.

[6]In Appendix C.1 we show that most samples of a reasoning step are not one or the other but both. This has implications for the intervention side effect discussed in §4.



(a) Results for Experiment 3

(b) Results for Experiment 4

Figure 2: Both results are for the Summation 2 step. (a) Intervention result when intervening at different layers. (b) Intervention results when intervening on the token-level inputs of the reasoning step, given different conditions of intervention on the attention pattern. *Experiment 1* shows a run without fixing the attention pattern (same as in Figure 1). *Intervened Attention* shows the mean result of fixing the attention pattern to each of 20 randomly sampled intervened runs. *Clean Attention* shows the mean result of fixing the attention pattern to 20 randomly sampled runs without any interventions.

diate computation, which are hence necessarily faithful. When there is an intervention, the model attempts to justify the correct answer by creating other causal dependencies that are unfaithful to the reasoning text. In §4, we discuss evidence of such behavior.

4 Intervening on the CoT Changes the Causal Mechanism

In this section, we show that interventions can change the causal relations they are intended to measure. A reasoning step may function as genuine intermediate computation under ordinary conditions, yet lose that role once it is perturbed as the model shifts to alternative pathways that still support the correct answer. This means intervention results do not necessarily reveal the mechanism the model uses in the wild. We test this possibility on the Summation 2 step across two experiments,[7] described below.

4.1 Experiment 3: Evidence from Layerwise Patching

In our experiments so far, we have made interventions directly on the tokens themselves. This measures the causal relations between two tokens, but it does not tell us what happens in between, as the model processes the information across layers and token positions. In this experiment, we focus on how the model responds to the intervention across layers. Specifically, we consider the possibility that the model detects the intervention in early layers and adapts accordingly, prior to further computation in middle and late layers.

If the model's mechanism changes in response to recognizing an intervention, the ability to adapt should be affected by the layerwise positioning of the intervention. That is, if an intervention occurs at too late a layer, the model might miss the opportunity to adjust its mechanism accordingly. To test this, we perform interchange interventions as in Section 3.1, but we now intervene at each intermediate layer rather than only at the input layer.

Figure 2a shows the proportion of original vs. counterfactual responses that result. We see that, indeed, there appears to be a "sweet spot" (in the middle layers 5-19) within which the intervention results in the model producing the counterfactual response. Interventions before and after this window both lead the model to primarily produce its original response.

This pattern suggests a mechanism in which the model uses early layers to evaluate the chain and detect possible problems or inconsistencies. If the intervention is made early, the model may subsequently adjust its mechanisms (e.g., its attention patterns) to avoid relying

[7] A few other steps showed similar patterns, though most not as pronounced. See Appendix D and Appendix E.2.

on the faulty chain. If, however, the intervention happens in a middle layer, after the model has committed to a mechanism but before it has committed to an answer, the intervention is more likely to result in the counterfactual response.[8]

4.2 Experiment 4: Evidence from Frozen Attention Patterns

We now consider how an intervention might affect the flow of information across layers. That is, we consider the scenario in which the model "notices" the intervention and, as a result, changes its attention mechanism to avoid the erroneous representation.

To test this, we first run the model on 20 withheld addition problems without any interventions and capture the attention patterns that result from these runs. We then run the model on the original 256 addition problems, each with one intervened reasoning step. Finally, we manipulate attention intervention in two conditions, comparing both to Experiment 1, where we did not intervene further, letting the model shift attention in response to the intervention. In the critical Clean Attention condition, we patch the frozen attention patterns from the clean runs on the withheld problems into the original problems. This essentially forces the model to adopt the kind of information flow it uses when there is no intervention. If indeed the model responds to intervention by "distrusting" an intervened token, we expect this frozen Clean Attention pattern intervention to result in the model relying on the counterfactual tokens as much as it would normally rely on the original tokens, leading it to output the counterfactual answer at a substantially higher rate than in Experiment 1.

The Intervened Attention condition controls for the effect of patching attention patterns from the 20 withheld addition problems to the 256 original ones. Here, we intervene at the token-level on the 20 withheld problems, freeze the attention pattern acquired from these intervened runs, and patch it back into the attention pattern of the model run on the original problems. This essentially forces the model to adopt the kind of information flow that results from an intervention. If this attention pattern allows the model to "distrust" the manipulation, then the results of the Intervened Attention condition should pattern with what we saw in Experiment 1, when the model was able to freely adjust its attention in response to the intervention. If, on the other hand, Experiment 1 and the Clean Attention conditions differ solely due to the effect of patching a frozen activation pattern, independent of the content of that pattern, then the Intervened Attention condition should differ from Experiment 1 as well.

Figure 2b shows the results of this experiment. We see that, indeed, when the model is presented with a counterfactually-intervened input but forced to use the attention pattern that it would have used on a normal input, the model produces the counterfactual answer 73.3% of the time. In contrast, when the model is allowed to adjust its attention pattern in response to the input, or is forced to use the attention patterns from other intervened runs, it produces the counterfactual answer only 21.1% and 14.7% of the time, respectively. This suggests that the model does recognize interventions in its inputs, and that the presence of such interventions is associated with an attentional shift.

4.3 Implications

The fact that causal relations change due to interventions is highly related to the finding from §3 that the models perform post-hoc justification. Since the model's training objective incentivizes it to arrive at the correct answer, it may respond to an introduced error by attempting arrive at the correct answer despite it, whether at the level of token generation or internal processing mechanisms. In the former case, we observe recovery surface to post-intervention reasoning generation shown in Table 1, while in the latter case the adaptive response may be hidden away in internal mechanisms, as in the shift in attention patterns observed in 4.2.

[8]We further conduct an experiment where we intervened at every layer *up to* a certain layer, instead of intervening only at one layer. It shows a very similar pattern, shown in Figure 5. This supports the conclusion that what matters is how the model processes the intervened token within the same residual stream, not somewhere else. See further discussion in Appendix D.

5 Discussion

When a reasoning model reports a chain of thought leading to its answer, is that chain how it actually got to that answer? Our results suggest a mixed picture. In Experiment 1, interventions on intermediate steps often did not affect the next step in the expected way. In Experiment 2, when allowed to continue generating after an intervention, models sometimes returned to the original answer, sometimes with additional confabulated reasons and sometimes without. At the same time, Experiments 3 and 4 show that the apparent effect of a CoT step depends on how it is tested: when we intervene on the CoT, models reduce their dependence on the intervened content, with early layers supporting detection of the intervention and altered attention patterns supporting adaptation to it. Taken together, these results suggest both that CoT mixes genuine intermediate computation with post-hoc justification, and that experiments intended to measure this mix can themselves change it.

Implications for Mechanistic Interpretability: Our results illustrate a challenge for AI interpretability. In interpretability, specifying the causal contribution of a hypothesized mechanism has often been treated as the gold standard both for showing that the mechanism exists and for understanding how it works. However, our results suggest that causality here is not static, and that it is not clear a model component's causal contribution can always be specified as a stable effect under counterfactual intervention. Rather than acting as a neutral probe of an existing structure, interventions may move the model into a different distribution (Makelov et al., 2024; Grant et al., 2025), with a different causal structure. As a result, intervention results need not reveal the mechanism the model uses on ordinary, unintervened runs. If the goal of mechanistic interpretability is to illuminate the internal structure of AI for the purposes of understanding, predictability, and control, this may require further mediation analyses of intervention side effects, or extensions of causal abstraction (Geiger et al., 2025) that account for the causal role of the measurement itself.

Reasoning, Rationalization, Faithfulness, and Trustworthiness: There is an active debate about whether CoTs should be treated as trustworthy explanations of a model's process. One useful way of framing this question is to distinguish causation from justification. In a trustworthy explanation, each step in a reasoning chain helps produce subsequent steps and also provides a reason to believe them (Fodor, 1985). For example, when someone adds 204 + 597 and shows their work, each intermediate step both causes the next step and justifies it.

But causation and justification can come apart. When asked to show their work for 4 + 7, a person may already know the answer by memory retrieval (Logan, 1988) and only then write the intermediate steps. In that case, the steps still justify the answer, but they do not describe the process that caused it; they are a rationalization. The converse dissociation is also possible: a process may causally generate the correct answer without offering a justification for it. This distinction matters because a strong skeptical view would hold that model reasoning can only ever be rationalization. Our results instead point to a more complex picture. Sometimes, CoTs appear to function as genuinely trustworthy explanations, in which causation and justification align. At other times, the model seems to justify an answer it had already effectively reached, or to depend on causal pathways not transparently represented in the CoT. At the same time, our findings suggest a limit on how confidently these cases can be distinguished, since the methods used to test the causal role of a CoT can themselves modify that role. The mere presence of a plausible CoT, therefore, does not provide strong evidence that it faithfully explains how the model arrived at its answer.

Model Training Incentivizes Both Justification and Intervention Side Effect: Our results can be understood in light of model training. Humans often produce justificatory or confirmatory reasoning, so models trained on human text should also learn such patterns. These tendencies may persist after RLHF post-training, since RLHF does not distinguish between causally utilized reasoning and plausible justification (Chang, 2026). Humans also make small errors, so models are implicitly trained to handle reasoning errors. The training distribution may therefore include genuinely utilized reasoning, justificatory reasoning, and erroneous reasoning, all of which a good next-word predictor must learn to navigate.

CoT is also a natural place to observe intervention side effects. Reasoning traces often contain redundant steps, whether stylistically or because they better satisfy training objectives (Chen et al., 2024; Chiang & Lee, 2024; Mao et al., 2025), and overdetermination by redundant causal influences creates the conditions for intervention side effects (Mueller, 2024). In this sense, the intervention side effects we observe can be understood as reflecting a tension between generating reasoning that looks good and producing the correct answer regardless.

6 Related Work

Early work on CoT often evaluated whether CoTs are plausible, well-formed, or internally coherent as explanations (Saparov & He, 2022; Ye & Durrett, 2022; Xia et al., 2025; Golovneva et al., 2023; Madsen et al., 2024). Tan (2023) suggests that CoT can be genuinely useful to the model, but Pfau et al. (2024) discovers that performance gains may arise partly from the extra computation enabled by longer outputs rather than from the verbalized steps themselves. Related work also suggests that models often generate unnecessary or redundant reasoning, even when they ultimately answer correctly (Chen et al., 2024; Chiang & Lee, 2024; Mao et al., 2025). More broadly, models can produce explanations that are influenced by shortcuts, biases, or latent factors not transparently reflected in the CoT (Turpin et al., 2023; Chen et al., 2025; Lindsey et al., 2025). Together, this literature shows that CoTs can look structured and explanation-like, but plausibility alone does not establish faithfulness.

Several papers intervene on CoT traces and measure the effect on the final answer (Lanham et al., 2023; Emmons et al., 2025). Others treat different sections of inputs—e.g., instruction, few-shot examples, reasoning trace, and answer—as causal variables and show that final answers often depend both on the original input and on the generated CoT (Dutta et al., 2024; Paul et al., 2024; Bao et al., 2025). Taken together, these results suggest that CoT does not uniquely mediate the answer, but instead coexists with additional causal pathways from the input, a form of *overdetermination*. At the level of individual reasoning steps, Dutta et al. (2024) uncovers attention heads that genuinely perform subtasks implied by the CoT, while Tan (2023) and Gao (2023) intervene on tokens to measure causal effect like we did in our experiments. Tan (2023) especially uses causal abstraction that resembles our Experiment 1. Together, they find evidence of genuine causal reasoning, but do not further analyze cases of post-hoc justification. Gao (2023) further finds overdetermination among reasoning steps and, along with Dutta et al. (2024), consider the possibility of intervention side effects, which we show in this paper.

Our methodological concern connects to a broader interpretability literature showing that interventions can produce misleading causal conclusions when the system compensates for disruption or when the probe itself activates alternative pathways (Mueller, 2024). In some cases *false negative* results can occur: a component looks non-causal under intervention because other components compensate for its disruption. In circuit-level studies, this appears as backup heads or related redundant mechanisms (Wang et al., 2022), and more generally as Hydra-like self-repair, where ablating one pathway recruits others that preserve behavior (McGrath et al., 2023). The opposite problem is also possible. Makelov et al. (2024) show that manipulating a representation that is not ordinarily causally important may activate a dormant pathway and change the output, creating *false positive* results. Emerging work explores ways to reduce damage to model representations caused by broad interventions (Zhang & Nanda, 2023; Grant et al., 2025), which our work complements by examining the distortions caused by interventions.

7 Limitations and Future work

We analyze a specific task to facilitate systematic interventions, leaving open how well the results generalize to other reasoning tasks. Such tasks could include more difficult formal tasks, as well as less defined tasks in natural language reasoning. Although it has been shown that CoT is likely more causally important with harder tasks (Emmons et al., 2025)—as utilizing the intermediate steps becomes more necessary—we suspect that post-hoc reasoning and intervention side effects can still occur with increased task difficulty, for

each individual step is significantly easier. Moreover, although we discover evidence for intervention side effects, we use special purpose experiments to better understand their role. Future work might develop more general interpretability frameworks that minimize, control for, or quantify intervention side effects and handle cases of redundant mechanisms.

References

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Guangsheng Bao, Hongbo Zhang, Cunxiang Wang, Linyi Yang, and Yue Zhang. How likely do LLMs with CoT mimic human reasoning? In Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, and Steven Schockaert (eds.), *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 7831–7850, Abu Dhabi, UAE, January 2025. Association for Computational Linguistics. URL https://aclanthology.org/2025.coling-main.524/.

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, 2024. Available at https://fbarez.github.io/assets/pdf/Cot_Is_Not_Explainability.pdf.

Edward Y. Chang. Right for the wrong reasons: Epistemic regret minimization for causal rung collapse in llms, 2026. URL https://arxiv.org/abs/2602.11675.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, et al. Do not think that much for 2+ 3=? on the overthinking of o1-like llms. *arXiv preprint arXiv:2412.21187*, 2024.

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Somani, 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.

Cheng-Han Chiang and Hung-yi Lee. Over-reasoning and redundant calculation of large language models. In Yvette Graham and Matthew Purver (eds.), *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 161–169, St. Julian's, Malta, March 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.eacl-short.15. URL https://aclanthology.org/2024.eacl-short.15/.

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

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. *arXiv preprint arXiv:2507.05246*, 2025.

Jerry A. Fodor. Précis of the modularity of mind. *Behavioral and Brain Sciences*, 8(1):1–5, 1985. doi: 10.1017/S0140525X0001921X.

Leo Gao. Shapley value attribution in chain of thought. https://www.lesswrong.com/posts/FX5JmftqL2j6K8dn4/shapley-value-attribution-in-chain-of-thought, 2023. Accessed: 2025-12-22.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal abstractions of neural networks, 2021. URL https://arxiv.org/abs/2106.02997.

Atticus Geiger, Duligur Ibeling, Amir Zur, Maheep Chaudhary, Sonakshi Chauhan, Jing Huang, Aryaman Arora, Zhengxuan Wu, Noah Goodman, Christopher Potts, et al. Causal abstraction: A theoretical foundation for mechanistic interpretability. *Journal of Machine Learning Research*, 26(83):1–64, 2025.

Olga Golovneva, Moya Peng Chen, Spencer Poff, Martin Corredor, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. ROSCOE: A suite of metrics for scoring step-by-step reasoning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=xYlJRpzZtsY.

Satchel Grant, Simon Jerome Han, Alexa Tartaglini, and Christopher Potts. Addressing divergent representations from causal interventions on neural networks. *arXiv preprint arXiv:2511.04638*, 2025.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamilė 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.

Jack Lindsey, Wes Gurnee, Emmanuel Ameisen, Brian Chen, Adam Pearce, Nicholas L. Turner, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. On the biology of a large language model. *Transformer Circuits Thread*, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/biology.html.

Gordon D. Logan. Toward an instance theory of automatization. *Psychological Review*, 95: 492–527, 1988. URL https://api.semanticscholar.org/CorpusID:10344934.

Andreas Madsen, Sarath Chandar, and Siva Reddy. Are self-explanations from large language models faithful? In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 295–337, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.19. URL https://aclanthology.org/2024.findings-acl.19/.

Aleksandar Makelov, Georg Lange, Atticus Geiger, and Neel Nanda. Is this the subspace you are looking for? an interpretability illusion for subspace activation patching. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Ebt7JgMHv1.

Minjia Mao, Bowen Yin, Yu Zhu, and Xiao Fang. Early stopping chain-of-thoughts in large language models. *arXiv preprint arXiv:2509.14004*, 2025.

Thomas McGrath, Matthew Rahtz, Janos Kramar, Vladimir Mikulik, and Shane Legg. The hydra effect: Emergent self-repair in language model computations. *arXiv preprint arXiv:2307.15771*, 2023.

Aaron Mueller. Missed causes and ambiguous effects: Counterfactuals pose challenges for interpreting neural networks. *arXiv preprint arXiv:2407.04690*, 2024.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. Show your work: Scratchpads for intermediate computation with language models, 2022. URL https://openreview.net/forum?id=iedYJm92o0a.

Debjit Paul, Robert West, Antoine Bosselut, and Boi Faltings. Making reasoning matter: Measuring and improving faithfulness of chain-of-thought reasoning. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 15012–15032, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.882. URL https://aclanthology.org/2024.findings-emnlp.882/.

Jacob Pfau, William Merrill, and Samuel R. Bowman. Let's think dot by dot: Hidden computation in transformer language models, 2024. URL https://arxiv.org/abs/2404.15758.

Abulhair Saparov and He He. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. *ArXiv*, abs/2210.01240, 2022. URL https://api.semanticscholar.org/CorpusID:252693237.

Juanhe (TJ) Tan. Causal abstraction for chain-of-thought reasoning in arithmetic word problems. In Yonatan Belinkov, Sophie Hao, Jaap Jumelet, Najoung Kim, Arya McCarthy, and Hosein Mohebbi (eds.), *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pp. 155–168, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.blackboxnlp-1.12. URL https://aclanthology.org/2023.blackboxnlp-1.12/.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems*, 36:74952–74965, 2023.

Constantin Venhoff, Iván Arcuschin, Philip Torr, Arthur Conmy, and Neel Nanda. Understanding reasoning in thinking language models via steering vectors, 2025. URL https://arxiv.org/abs/2506.18167.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf.

Shijie Xia, Xuefeng Li, Yixin Liu, Tongshuang Wu, and Pengfei Liu. Evaluating mathematical reasoning beyond accuracy. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(26):27723–27730, Apr. 2025. doi: 10.1609/aaai.v39i26.34987. URL https://ojs.aaai.org/index.php/AAAI/article/view/34987.

Xi Ye and Greg Durrett. The unreliability of explanations in few-shot prompting for textual reasoning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=Bct2f8fRd8S.

Fred Zhang and Neel Nanda. Towards best practices of activation patching in language models: Metrics and methods. *arXiv preprint arXiv:2309.16042*, 2023.

A Appendix A. Definition of Justification, Faithful Explanation, and Causal Reasoning

A.1 Informal Definitions

Here are informal definitions we use:

- Task: A task is defined by a set of possible inputs and a function from inputs to outputs.
- Subtasks of a task: A set of tasks that can be combined to perform the task. There can be subtasks within subtasks.
- Instance/Run of a Task: A task with a specific input. Often simply called a task.
- Solution: A process that maps a specific input to the appropriate output according to a task.
- Description (of a solution): A natural language description of a solution.
- CoT: Defined by convention. It is required to be a description that occurs between the input and output of a task.
- Step: Given a subtask of a task, a step is a description for the subtask within a description for the task. The subtask is often required to not have its own subtask.
- Causal: A description is causal if it determines the output.
- Justification/Explanation: A description that is not causal. The term "post-hoc" is often used to emphasize the lack of causality.
- Causal Step: A step that is causal.
- (Causal) Reasoning: A CoT where every step is causal. This can be a graded notion based on how causal each step is.
- Faithfulness: Whether or not a description accurately describes the model's computational process from the inputs to the output.

Given these descriptions, the goal of §3.1, which is to determine whether a CoT is (causal) reasoning or (post-hoc) justification, becomes testing if each step of the CoT is causal. Our methodology follows naturally. It also follows that a causal reasoning is necessarily a faithful CoT while a justification could be or could be otherwise.

A.2 Faithfulness and Causal Reasoning through Causal Abstraction

To measure faithfulness, we need to measure how accurately a natural language text describes a computational process. To do so, we employ causal abstraction (Geiger et al., 2025) that defines the faithfulness of a high-level causal model (often described with a Directed Acyclic Graph (DAG)) to a low-level computational process. What is left to do then is to map a natural language text to a DAG, which we do below, similar to Tan (2023).

We assume that each well-formed reasoning text naturally implies a (set of) causal models. For instance, we can view the text in Figure 3a to imply the DAG shown in Figure 3b. Given this DAG, one can measure the faithfulness using the process described in Geiger et al. (2025).

Now, a causal reasoning is not only faithful but each of its steps also has to be causal with respect to the subtask it performs. This can be captured by aligning the tokens in the CoT with the corresponding variables in the DAG. In fact, causal abstraction involves a specification of a mapping from low-level computational parts to the high-level variables (partition Π and mapping π in Geiger et al. (2025)). For causal reasoning, we can take the tokens as the low-level variables and map them to the appropriate variables in the DAG. For instance, we would map the first instance of the token `111` to the variable with ID 11, and map the second instance of the token `400` to the variable with ID 18.

B Appendix B. Experimental Setup and Definition of Steps

For reproducibility, we will provide the GitHub repository after the review process.

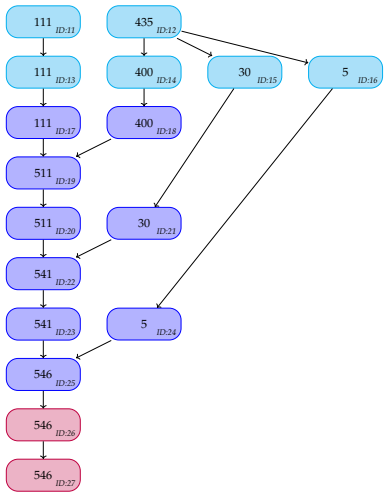
CoT:
We need to add...

111 + 435 = 111 + 400 + 30 + 5

= 111 + 400 = 511, 511 + 30 = 541, 541 + 5 = 546.

So the sum is 546. In final, output only "546".

(a) A sample CoT. Linebreak was added for legibility.



(b) High-level causal model implied by the CoT to the left.

Figure 3: The two subfigures illustrate the process of deriving a high level causal model from a natural language text. For the graph on the right, the variables are colored to distinguish *Breakdown*, *Summation 1, 2, 3*, and *Final 1, 2*. Each variable is marked with an ID number for future reference.

B.1 Model and Prompt Setup

Table 2 shows the models and prompts we used for the experiments. The main paper and most of the appendix discusses results from the GPT-OSS-20B model with the "Stepwise" reasoning format prompting. Results from other model-prompting setup is shown in §E.

GPT-OSS-20B

Reasoning	Developer Prompt	User Prompt
Stepwise	# Instruction: In analysis, add two numbers stepwise. In final, output only the sum.	What is 123+456
Column	# Instructions\nYou are a stepwise adder that reasons concisely. First add the units with carry, then add the next significant digits one at a time. Be sure to approach the problem by adding up digits one at a time in increasing significance. Once reasoning has concluded, give the user only the final answer.	What is 123+456?
Default	*None*	What is 123+456?

DeepSeek-R1-Distill-Llama-8B

Name	Developer Prompt	User Prompt
Default	*None*	What is 123+456? Do not tell me how to do it, just give me the answer. For example, "The answer is 397".

Table 2: Summary of prompting strategy for reproducibility.

GPT-OSS-20B Stepwise Prompt Sample Generations
The user asks: "What is 693+214?" The instruction from the developer says: "In analysis, add two numbers stepwise. In final, output only the sum." So I need to do the addition stepwise in the analysis, and then in the final output, just output the sum. So I need to do 693 + 214. Let's do stepwise: 693 + 214 = 693 + 200 + 14 = 893 + 14 = 907. Wait, check: 693 + 200 = 893. 893 + 14 = 907. Yes. So the sum is 907. In the final, I should output only "907".<|end|><|start|>assistant<|channel|>final<|message|>907<|return|>
The user asks: "What is 852+664?" The instruction from the developer says: "In analysis, add two numbers stepwise. In final, output only the sum." So I need to do the addition stepwise in the analysis, and then in the final output, just output the sum. So I need to show the stepwise addition: 852 + 664. Let's do it: 852 + 664 = 852 + 600 + 60 + 4 = 852 + 600 = 1452, 1452 + 60 = 1512, 1512 + 4 = 1516. Or do column addition: 852 + 664 = 1516. So the sum is 1516. In final, output only "1516".<|end|><|start|>assistant<|channel|>final<|message|>1516<|return|>
The user asks: "What is 984+776?" The instruction from the developer says: "In analysis, add two numbers stepwise. In final, output only the sum." So I need to do the addition stepwise in the analysis, and then in the final output, just output the sum. So I need to do 984 + 776. Let's do stepwise: 984 + 776. 984 + 700 = 1684. 1684 + 70 = 1754. 1754 + 6 = 1760. So the sum is 1760. In the final, I should output only "1760".<|end|><|start|>assistant<|channel|>final<|message|>1760<|return|>

Table 3: Randomly sampled generations by GPT-OSS-20B given the *Stepwise* developer prompt.

B.2 Standardizing the CoT

In Table 3, we show sample generations from the *Stepwise* prompt.

As you can observe, they somewhat follow a similar trajectory. We hence standardize the CoT, shown in Table 4, which also shows the standardized CoT for other prompts. For other prompts, the generations are actually more standardized. This is because, for GPT-OSS-20B, the column prompt contains more specification and the default prompt makes the model spit out the answer without any meaningful steps, and DeepSeek-R1-Distill-Llama-8B is heavily trained on mathematical tasks such that it always follow a very specific line of reasoning.

B.3 Dataset

Given the prompts and the standardized CoTs, we now have to create 256 variations with different numbers. Note that each variation, called a *base prompt*, also has to come with an alteration of it, called the *source prompt* from which we will acquire the token or activation to patch into the base prompt.

For the *Default* prompt, we simply sample two random three-digit numbers for the base prompt, and two more three-digit numbers to create the source prompt. For the *Stepwise* prompt, we make sure the second addend has no zeros in its digits, because it would make an awkward addition of 0 in its steps (in natural generations, the model would skip a step if the digit is 0). For the source prompt, we also only vary the hundreds digit of the second addend compared to the base prompt. This is so that each step only has one input that varies, which facilitates the calculation of the counterfactual output. For the *Column* prompt, we make sure both addends have no zeros in their digits, and we also only vary the units digit of the second addend for the source prompt. The rationale is the same with that for the *Stepwise* prompt. The first three rows of the dataset is shown in Table 5.

GPT-OSS-20B

Prompt	Standardized CoT
Stepwise	<|start|>user<|message|>What is 204+443?<|end|><|start|>assistant<|channel|>analysis<|message|>The user asks: "What is 204+443?" The instruction from the developer says: "In the analysis, add two numbers stepwise. For the final output, output only the sum." So we need to do the addition stepwise in the analysis, and then in the final output, just output the sum. So I need to do stepwise addition in analysis, then output only the sum in final. So in analysis, I will show the stepwise addition: 204 + 443. Let's do it: 204 + 443 = 204 + 400 + 40 + 3 = 204 + 400 = 604, 604 + 40 = 644, 644 + 3 = 647. So the sum is 647. In final, output only "647".<|end|><|start|>assistant <|channel|>final<|message|>647<|return|>
Column	<|start|>user<|message|>What is 214+491?<|end|><|start|>assistant<|channel|>analysis<|message|> The user asks: "What is 214+491?" The developer instructions: "You are a stepwise adder that reasons concisely. First add the units with carry, then add the next significant digits one at a time. Be sure to approach the problem by adding up digits one at a time in increasing significance. Once reasoning has concluded, give the user only the final answer." So we need to do stepwise addition: units: 4+1=5, write 5 carry 0. Tens: 1+9=10, write 0 carry 1. Hundreds: 2+4=6 plus carry 1 = 7. So result 705. Provide only the final answer. So output: 705.<|end|><|start|>assistant<|channel|>final<|message|>705<|return|>
Default	<|start|>user<|message|>What is 10+24? <|end|><|start|>analysis<|message|>The user asks: "What is 10+24?" It's a simple arithmetic question. The answer is 34. The user likely expects a straightforward answer. There's no trick. So respond with "34". Possibly add a brief explanation. But the user didn't ask for explanation. Just answer. So answer: 34.<|end|><|start|>assistant <|channel|>final<|message|>34<|return|>

DeepSeek-R1-Distill-Llama-8B

Prompt	Standardized CoT
Default	<|User|>What is 204+243? Do not tell me how to do it, just give me the answer. For example, ""The answer is 397"".<|Assistant|><think> I need to add 204 and 243. First, I'll add the hundreds place: 200 + 200 equals 400. Next, I'll add the tens place: 0 + 40 equals 40. Then, I'll add the units place: 4 + 3 equals 7. Finally, I'll combine these results: 400 + 40 + 7 equals 447. </think> The answer is $\boxed{447}$.

Table 4: Standardized CoT to facilitate intervention experiments for different models and prompting. We create 256 CoTs for each model-prompting setup by swapping the numbers within the fixed CoTs.

Prompt	Base Numbers	Source Numbers
Default	204+443	281+711
	233+919	486+485
	126+653	451+272
Stepwise	204+443	204+243
	410+452	410+552
	130+675	130+275
Column	214+491	214+490
	484+145	484+143
	420+491	420+494

Table 5: Datasets for different prompting. Base Numbers are used to create the CoT we study, and Source Numbers are used to acquire tokens or activations to patch into the base CoT.

B.4 Definition of Steps

In §2 we defined steps that we reported results for throughout the paper. This was an expository choice to convey the idea that there are multiple interrelated causal relations to test to determine the faithfulness of a CoT. Those steps were defined based on how a person would intuitively divide up the CoT into elementary chunks.

To be more rigorous, we would need to define a step by the subtask it is performing. Under the causal abstraction framework, we can define a subtask based on a causal model thus: a subtask is a computation of one causal variable based on all the causal variables it is based on.

Many of the steps we define—Summation 1, 2, 3 and Final 1, 2— already follow this definition. However, this definition means that there are more steps to check, including those for computing variables with ID 20 and 23. We do so below in §C.2.

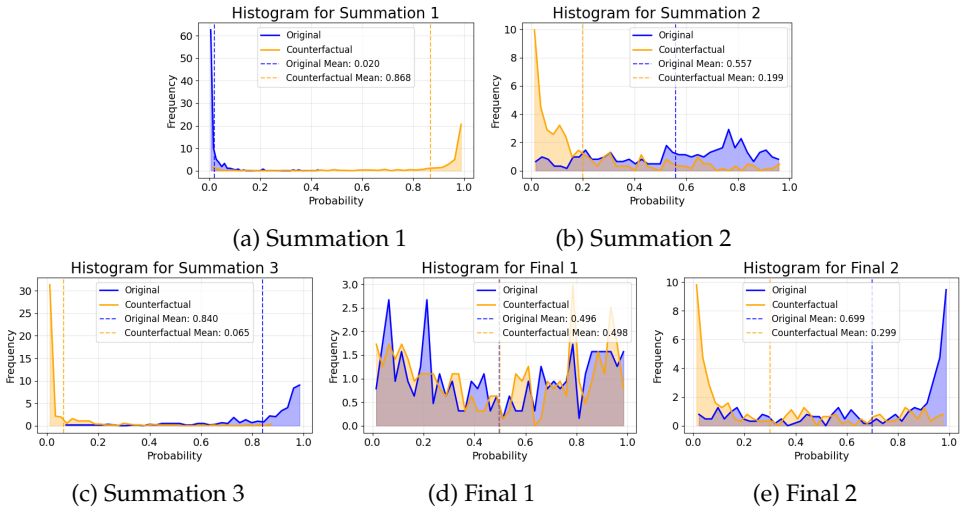
C Appendix C. Supplemental Results for §3

C.1 Histogram of Output Probabilities from Experiment 1

In Figure 1 in §3.1, we showed the frequency of output types (counterfactual, original, or other) based on the top predicted token given an intervention. But only looking at the top token might hide a lot of information from us. For instance, if the counterfactual output always has 70% probability and the original output always has 30%, we would get 100% counterfactual output, even though the model still always calculates the original output too.

Below, we show the histogram for output probabilities across 256 runs at each step. We overall find that (1) the mean probability and the frequency in Figure 1 are roughly proportional, and, more importantly, (2) mechanisms that calculate the counterfactual output and others that calculate the original output are often simultaneously active. The latter point is the most poignant in Figure 4d where the frequency of the original output and the counterfactual output are roughly equal at each probability.

In other words, it is not that a reasoning step is sometimes genuine and sometimes a justification, but it is most often both. It's just that sometimes it is more of a causal reasoning, and at other times it is less so. Then intervention side effect can be seen as simply lowering the degree to which a step is genuine. Moreover, the co-existence of both mechanisms and the change in relative weighting between them is coupled, in the sense that the same training pressure to get the output correct would incentivize both the creation of mechanisms for predicting the original answer, but also for making the model raise their importance when there is an intervention.



(a) Summation 1 (b) Summation 2

(c) Summation 3 (d) Final 1 (e) Final 2

Figure 4: Histogram for output probabilities across 256 runs at different steps.

C.2 Other Causal Relations from Experiment 1

In Figure 1 in §3.1, we only showed the causal relations between tokens that are expected to be related. For instance, we expected the blue token below to be related to the yellow token:

> **CoT:** We need to add...111 + 435 = 111 + 400 + 30 + 5 = 111 + 400 = 511, 511 + 30 = 541, 541 + 5 = 546. So the sum is 546. In final, output only "546".

Given that the causal dependence was weak, we can ask what else the blue token depends on. For instance, what if it is instead dependent on the pink token below?

> **CoT:** We need to add...111 + 435 = 111 + 400 + 30 + 5 = 111 + 400 = 511, 511 + 30 = 541, 541 + 5 = 546. So the sum is 546. In final, output only "546".

Or what if it depends on whatever agrees with the addends of the previous step, considering any of the two '511' tokens could be typos?

> **CoT:** We need to add...111 + 435 = 111 + 400 + 30 + 5 = 111 + 400 = 511, 511 + 30 = 541, 541 + 5 = 546. So the sum is 546. In final, output only "546".

or

> **CoT:** We need to add...111 + 435 = 111 + 400 + 30 + 5 = 111 + 400 = 511, 511 + 30 = 541, 541 + 5 = 546. So the sum is 546. In final, output only "546".

We test many combination and report the result in the table below:

Intervened Tokens	Counterfactual Output Rate
511_1	9.4%
511_2	21.1%
511_1 & 511_2	100.0%
$111{+}40_2$	0.0%
$111{+}40_2$ & 511_1	0.4%
$111{+}40_1$ & $111{+}40_2$ & 511_1	19.5%
$111{+}40_2$ & 511_2	52.7%
$111{+}40_1$ & $111{+}40_2$ & 511_2	58.8%

Table 6: The counterfactual output rate for variable with ID 22 (the '541' token) when intervening at different tokens. 511_1 and 511_2 refers to the first and second instance of the number '511', respectively, and $111{+}40_1$ and $111{+}40_2$ refers to the first and second instance of the sequence '111+40', respectively.

This set of results is very interesting. If the two instances of '511' agree, then the model relies on them. If they disagree, it seems to rely on a third source most of the time, which has to be before the Breakdown step (observe that the row $111{+}40_1$ & $111{+}40_2$ & 511_1 and 511_2, which accounts for Breakdown, Summation 1, and Summation 2, sum up to 40.6%). This means it is likely that the models calculates what the intermediate sum has to be based on the two original addends and what their sum (i.e. the correct final sum) has to be. Then it would be a clear case of justification.

Next, we test intervention results for steps we did not explicit cover in the main paper, as we discussed in §B.4. Those are the "copying" steps where the model simply copies a previous number, shown below (the Final 1, 2 steps are also "copying" steps but we treated them separately):

> **CoT:** We need to add...111 + 435 = 111 + 400 + 30 + 5 = 111 + 400 = 511, 511 + 30 = 541, 541 + 5 = 546. So the sum is 546. In final, output only "546".

We report that the counterfactual output rate for these pairs of tokens is 100%.

Finally, we actually have not reported at all if the final output for the user is dependent on the CoT. We report here that, when only the output of the Final 2 step is intervened, the CoT always outputs the original answer. However, if both Final 1 and Final 2 are intervened, it outputs the counterfactual answer 99.2% of the time. This behavior or relying on two numbers only when they agree resembles the Summation 2 step.

C.3 Annotations for Experiment 2

In this section, we describe the methodology we used to annotate post-intervention generation samples, shown in §3.2 in the main paper.

Manual Annotation Process: There are total 3072 generations from 256 additions (i.e. numbers being added) and 12 intervening positions. Out of those, we choose 5 additions randomly and take the 12 generations after 12 intervening positions from each of them. This totals up to 60 samples.

Given these 60 samples, two annotators annotated them freely without the labels chosen in advance. One annotator annotated the samples with four total labels, and the other did so with five. They then convened and agreed to use the four labels. Given these four labels, there was one case of disagreement, but it was deemed a mistake by one annotator. Hence, there was complete agreement between the two annotators on the 60 samples.

LLM Annotation Process: We used Claude Sonnet 4.6 to label all 3072 generations. We adjusted the prompt until it agreed with the two human annotators on 57 out of 60 selected samples, and the prompt is shown below:

I want you to annotate a reasoning chain generated by a language model where a researcher has introduced an error to test how the model handles it. Below you will see the reasoning chain up to the error, the natural continuation without an error, the error introduced by the researcher, and the model's continuation provided after the error. I will also provide you the correct and incorrect final answers, so that you can easily detect the 'Override' cases by checking whether the correct answer simply overrides the incorrect answer.

Use the following steps in order to annotate the reasoning chain:
1. Output "Follow Through" if the final answer is incorrect.
2. Output "Correction" if the post-error starts off by immediately acknowledging the error (e.g. "wait" or a question mark).
3. Output "Override" if the Post-error continuation has this sequence: "= incorrect_final_answer. So the sum is correct_final_answer?" It may still acknowledge the error afterwards, but not before the sequence.
4. Output "Recovery" if the Post-error continuation gets to the correct answer by adding a step (often close to the beginning) that is not found in the Natural continuation. For instance, it might add an extra number or subtract a necessary quantity to get the reasoning steps back on track. It may still acknowledge the error, but not before the extra step.

Please annotate the reasoning chain with the appropriate label. Output only the label (e.g. "Correction", "Follow Through", "Override", or "Recovery"):
Prefix: reasoning_chain
Natural continuation: natural_continuation
Error: error
Post-error continuation: model_continuation
Correct final answer: correct_final_answer
Incorrect final answer: incorrect_final_answer

Then, we used the prompt to annotate the rest of the samples, and we reported the result in Table 1.

Implications on the Faithfulness Metric of Lanham et al. (2023): We re-emphasize a point we made in a footnote in §3.2. Introducing an error through an intervention might alter the subsequent reasoning trajectory completely. If so, even if we find out whether this new trajectory is a causal reasoning or a post-hoc justification, we cannot conclude if the same is true for the reasoning trajectory that would have occurred without the intervention. In other words, just because a causal variable had an effect in a new causal structure, it does not mean it always has to have an effect in other causal structures it is part of. In some sense, we should really view "change in the reasoning trajectory" as an intervention side effect because it changes the causal structure the CoT implies.

Lanham et al. (2023) measures the unfaithfulness of CoT by seeing how often the model still arrives at the original final output despite an error introduced in the steps. Since the reasoning trajectory could change entirely, we argue that this metric does not measure CoT unfaithfulness of non-intervened CoT. But could it still tell us about the general tendencies of the model to causally reason or post-hoc justify? We argue that the metric of "arriving at the correct answer at the end" is still flawed. For instance, both *Correction* cases and the *Follow Through* suggest causal reasoning (faithful CoT), but one arrives at the correct answer while the other does not.

D Appendix D. Supplemental Results for §4

D.1 Layerwise Intervention

In §2a, we intervened at each hidden layer of the model, and found that intervening after layer 6 had a much higher counterfactual output rate than intervening earlier. We concluded

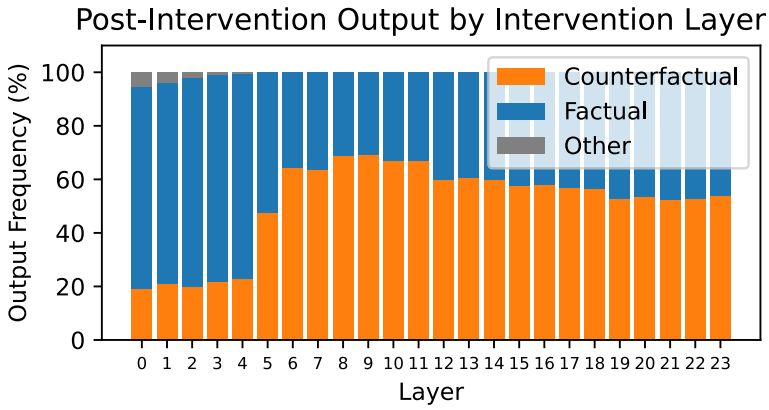


Figure 5: The result of intervening on all layers up to a certain layer (for the Summation 2 step). The result is very similar to Figure 2a.

that the model must be somehow marking that the token has been intervened on before layer 6.

There are two ways that intervening late could be important. The first possibility is that intervening late keeps the early layers clean (non-intervened), and that this alters the mechanism used. One hypothesis is that later token positions look at earlier token positions at early layers to detect errors. If this is the case, it is important that the early layers are clean. The second possibility is that intervening late could keep the later layers clean by patching in non-intervened representations into the residual stream. One hypothesis is that the residual stream of the intervened token marks "intervened-ness" around layer 4 or 5, and hence patching in after that would keep the residual stream free of this marking.

If we patch in hidden activations *up to* a certain layer instead of only at the layer, we can distinguish between these two hypotheses. If the first hypothesis is the case, all interventions should lead to a low counterfactual output rate. If the latter hypothesis is the case, we should see a similar pattern to Figure 2a where patching up to later layers raises the counterfactual output rate. The result that confirms the latter hypothesis is shown in Figure 5.

D.2 Frozen Attention Pattern Patching

We first describe further details of the experiment.

Getting the Attention Patterns: We have 256 samples of three-digit addition problems. We sample 20 more problems and acquire the CoT from the standardized formant shown in Table 4. We then create intervened copies of them where we intervene at a specific step. We run forward passes on those CoTs and capture their attention patterns, giving us 20 clean attention patterns and 20 intervened attention patterns.

Patching the Attention Patterns: We have 256 samples of three-digit addition problems. For each of them, we perform a forward pass with an intervention on a specific step while patching in one of the 20 clean attention patterns or 20 intervened attention patterns. This means, the 'Clean Attention' runs in Figure 2b contains 256·20 total runs, and the 'Intervened Attention' stack does so as well.

Why patch in attention patterns from 20 separate runs? An alternative would have been to simply run each of the 256 samples without the intervention and cache the attention pattern, and patch the clean attention pattern from each run to its intervened run. However, if we do so, we do not know if the resulting change in behavior results from the inherent difference between clean and intervened attention patterns, or if it is from the attention pattern patching confusing the model. Hence we patch in attention patterns from other

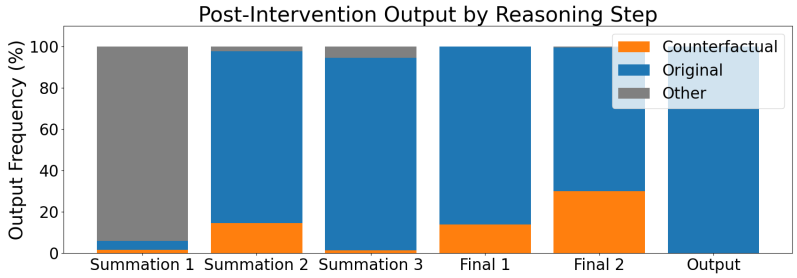


Figure 6: Intervention results for each step while patching in attention patterns from 20 sampled intervened runs.

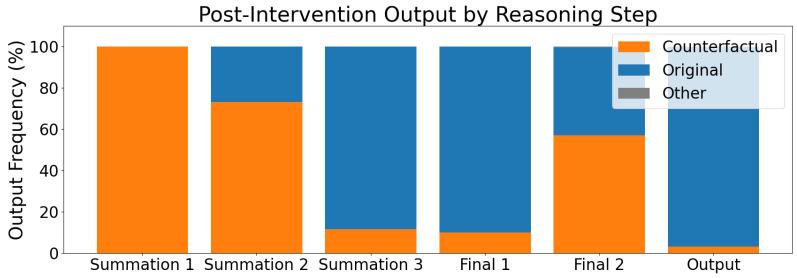


Figure 7: Intervention results for each step while patching in attention patterns from 20 sampled clean runs.

addition problems to each sample. We ran 20 of them because each attention pattern seems to be different in that they might use different heads for similar tasks. Hence we average over them to observe the general difference between clean and intervened attention patterns.

In the main paper, we showed frozen attention pattern patching results only for the Summation 2 step. We show the results for each step with Figures 6-7.

We find that patching in clean runs generally tends to raise the counterfactual response rate, meaning the reasoning is more genuine "in the wild" and less so under interventions. There are some anomalies with Figure 6 for the Summation 1 and Final 1 steps. For Summation 1, we qualitatively find that the model generally gets really thrown off and output random numbers similar to either the factual or the counterfactual answers. For Final 1, it is odd that counterfactual output rate for both Figure 6 and Figure 7 is lower than that for Figure 1. We performed further analysis, and it appears that the Final 1 step depends on a variety of sources of the original answer. Hence, we hypothesize that attention pattern patching generally damages mechanisms (by the mismatch between QK and OV circuits), and since there are more mechanisms for justifying the original answer than performing causal reasoning, the original answer tends to prevail.

D.3 Specific Mechanistic Changes Before and After the Intervention

We made some additional investigation into what is happening to the model before and after the intervention at a more microscopic level. The change seems to be distributed across many model components (e.g. attention heads) and also idiosyncratic to each of the 256 samples, and hence require more careful analyses.

One interpretable result we did find is that there is a linear direction that corresponds to something like "doubt" that we can remove from the intervened residual stream to recover the behavior "in the wild." We describe the experiment here.

Logic: Experiment 3 seems to suggest that the fact of being intervened is marked on the intervened residual stream somewhere between layers 4 and 5. If so, we should be able to

remove such marking and trick the model to think that there was no intervention. In this experiment, we look for a vector that could be the marking and subtract it from the residual stream. There are two vectors we test:

- **"Doubt" vector:** We obtain a vector that increases backtracking in reasoning from Venhoff et al. (2025) and take it to represent the direction of "doubt." We hypothesize that the model marks intervention by lowering confidence or trustworthiness of the residual stream. Hence, if we recover trustworthiness by subtracting the doubt vector, we expect the model to act as if there was no intervention.
- **Mean difference between clean and intervened residual streams:** We obtain 120 samples of clean and intervened residual streams, and take a difference between their means.

Results: Subtracting the doubt vector at layer 12 brings the counterfactual response rate up to 56.6% from 21.1%. This degree of causal dependence is much closer to the one found in attention freezing experiment or layer-wise patching experiment, meaning the model could be marking intervention with lack of confidence.

Subtracting the mean difference vector does not work as well. The vector itself is small (2-digit norm as compared 3- and 4-digit norms for an average residual stream activation), which means there is not a clear linear difference between clean and intervened residual streams. However, if we subtract a scaled up version of the vector, it does increase counterfactual response rate to 30.86%, which is around 10% higher than the baseline.

As controls, adding the vectors instead of subtracting them lowers the counterfactual response rate, and subtracting random vectors of similar size do not have any meaningful effect.

We find these results to be inconclusive as to how exactly the model marks the state of being intervened. Whereas confidence does seem to modulate how much the model trusts a residual stream, an average intervened residual stream from our experiment itself does not have lower confidence than clean ones (the linear difference between them is small and the cosine similarity between the difference and the "doubt" vector is close to 0). In general, there not being a strong linear difference between clean and intervened residual streams suggests that attention pattern shifts are not based on a linear representations.

E Appendix E. Results for DeepSeek-R1 and Prompt Setup

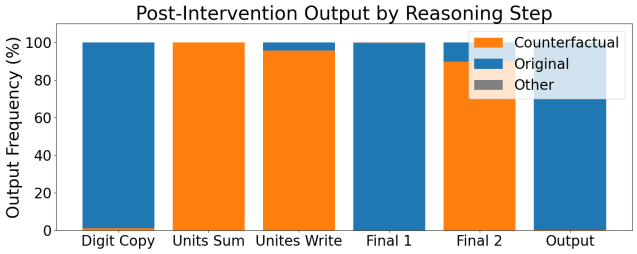
E.1 §3 Results

We replicate Experiment 1 for different model-prompting setups and report the results in Figure 8. We hypothesize that DeepSeek-R1-Distill-Llama-8B perform genuine causal reasoning significantly more often because its training regime heavily involves solving math problems.

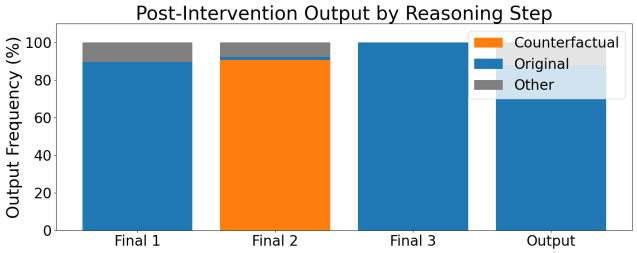
In general, each step is almost entirely causal reasoning or entirely post-hoc justification. However, this does not mean that this is the case "in the wild." In the next subsection, we show a case of intervention side effect for one of the settings.

E.2 §4 Results

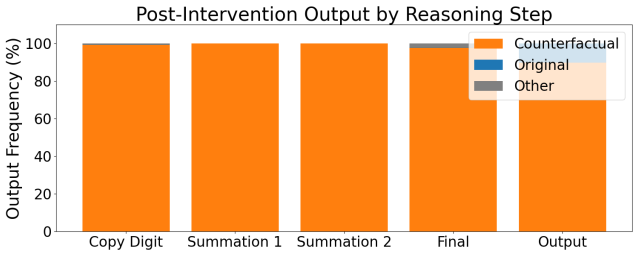
In Figure 9, we report layerwise intervention results for GPT-OSS-20B with the *Default* prompt, where we found a similar pattern to the *Stepwise* prompt.



(a) Token intervention results for each step in CoT for GPT-OSS-20B with the *Column* prompt.



(b) Token intervention results for each step in CoT for GPT-OSS-20B with the *Default* prompt.



(c) Token intervention results for each step in CoT for DeepSeek-R1-Distill-Llama-8B with the *Default* prompt.

Figure 8: Experiment 1 Results for various model-prompting setup. Most steps are close to being either entirely causal reasoning or entirely justification.

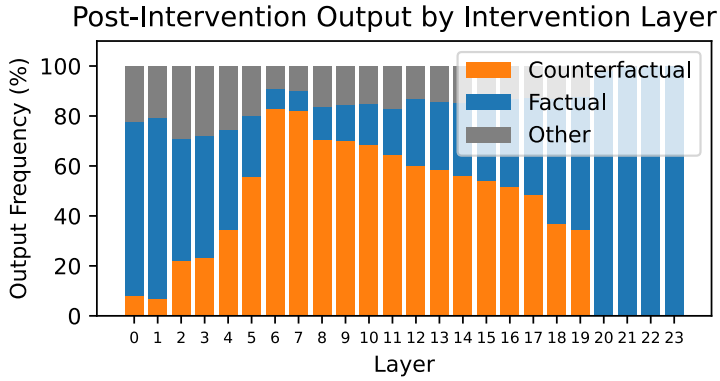


Figure 9: The result of intervening on each hidden layer for the Final 1 step of the *Default* prompt. The result is similar to Figure 2a.