

The Trace Is Not the State

A Counterfactual Test of What Controls Multi-Digit Addition

Tanmay Patil

Walchand Institute of Technology, Solapur, India

September 2026

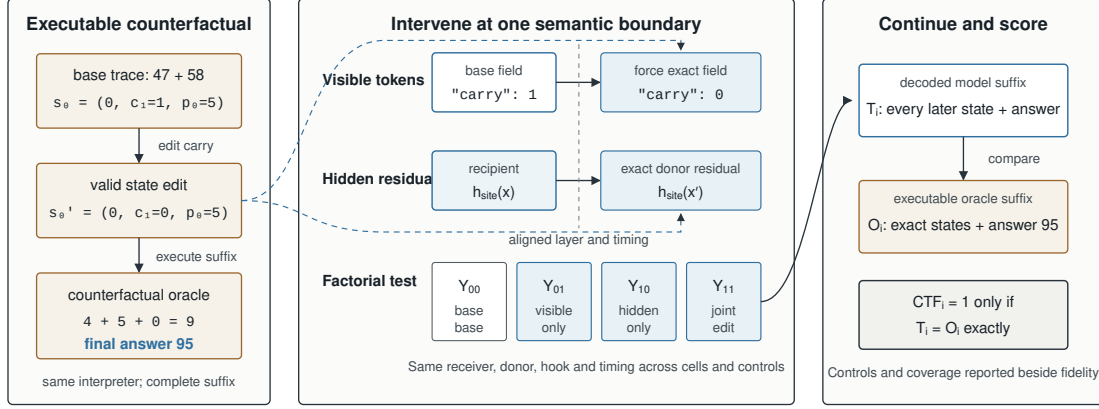
Abstract

A reasoning model can print **carry the one** while never having computed a carry. I specify an experimental protocol for testing whether an intermediate state in multi-digit addition has causal influence through hidden activations, visible reasoning tokens, both, or neither. Each arithmetic trace is generated by an executable interpreter and records the digit position, carry, partial result, next transition and final answer. The test defines a valid semantic counterfactual, replaces a complete donor residual vector in hidden arms, and checks whether the remaining trace follows that counterfactual computation. Hidden-only, visible-token-only and joint interventions are compared; the hidden and joint arms are controlled by shuffled-donor, unrelated-donor and same-norm random vectors, with hidden patches applied at three positions around state-token emission. Scoring uses full-trace counterfactual transition fidelity, nuisance invariance, repair, donor leakage and collateral change across separately held-out in-distribution (IID) and out-of-distribution (OOD) splits. The protocol fixes a prospective floor of 384 independent complete program clusters in every applicable evidence family; smaller samples remain descriptive and indeterminate. The collateral diagnostic is reported without a pre-specified threshold, so version 1 makes no selectivity claim from it. Attribution methods may rank candidate locations, but every retained location must pass an exact activation intervention. The paper reports no new model results.

Keywords: mechanistic interpretability; chain of thought; causal abstraction; activation patching; multi-digit addition; counterfactual intervention

Scope and evidential status

This manuscript specifies an experimental protocol; it contains no model results. Section 5 summarises prior work; none of those findings is presented as a replication. The design keeps discovery, calibration, validation and final data separate, and interrupted, invalid, pilot or smoke runs are classified as diagnostic records rather than scientific results. Methods discussed only in the literature review are outside the specified implementation.



Design schematic only; no model output is shown.

Graphical abstract. An executable carry edit defines the counterfactual oracle. Visible state tokens and hidden residual replacements are assigned in four factorial cells, after which the complete decoded suffix is compared with the oracle; a visible edit can alter later hidden states. The diagram states the design and contains no model output.

1. Introduction

Discussions of chain of thought often conflate generated tokens, the hidden computation that changes the model’s state, and the explanation a reader assigns to those tokens. They may agree on a particular example, but agreement cannot be assumed. A language model can use a printed numeral as temporary memory, retain a richer quantity in its residual stream, or generate an explanation after the answer-relevant computation has already occurred.

Multi-digit addition gives a useful setting in which to separate these possibilities. Its intermediate state can be specified exactly. After each column, the state contains the digit position, the carry into the next column and the partial result. A deterministic interpreter gives the next state and the final answer. This permits a direct counterfactual question: if the carry or partial result is replaced by a different valid state, does the model continue with the arithmetic implied by that replacement?

Attention plots, probes and attribution scores can help locate candidate representations, but they do not by themselves answer this question. A probe may decode information that later computation does not use. A component may appear unnecessary because another path repairs its ablation. A large patch may also create an activation outside the distribution on which the model was trained. For this reason I keep discovery and causal validation separate in this study. Attribution ranks candidate sites on discovery data; exact interventions test those sites on untouched validation data.

I cross four intervention conditions (hidden-only, visible-token-only, joint and no semantic edit) with shuffled donors, unrelated donors and same-norm random vectors as controls. I apply each hidden patch at three points: immediately before the state token is emitted, immediately after it is emitted, and before the next arithmetic operation. This timing is required because visible text can become part of the input to later token generation. A hidden edit made before emission and the same edit made after emission do not test the same causal path.

Prior results point in different directions. Controlled models can learn compact algorithms for modular arithmetic and synthetic iteration tasks (Nanda et al., 2023; Cabannes et al., 2024). Studies of pretrained models have found small causal components for restricted numerical

behaviours (Hanna et al., 2023; Bertolazzi et al., 2025), as well as distributed arithmetic heuristics and redundant reasoning paths (Dutta et al., 2024; Nikankin et al., 2025). Separately, interventions on generated rationales show that visible explanations can be causally important on some tasks and unfaithful on others (Turpin et al., 2023; Lanham et al., 2023; Y. Chen et al., 2025). These studies motivate the design, but none settles whether an exact addition state has authority through hidden activations, visible tokens, both, or neither.

Here, program-like behaviour means that a state edit produces the expected counterfactual transition across the declared evaluation distribution. The target may be a distributed representation rather than a unique neuron or attention head. I measure localisation and editability separately because redundancy, repair and distribution shift can change intervention outcomes. Visible tokens may provide external memory, routing cues, incomplete reports or post-hoc explanations, even within one trace.

I define the intervention test, discuss the representation and repair problems that affect it, and specify an executable study with multi-digit addition as the primary test case. Section 2 reviews the required background. Sections 3 and 4 define the causal criterion and evaluation method. Section 5 summarises directly relevant empirical work. Sections 6 to 9 present the proposed study, editing limits, interpretation and limitations.

2. Background

2.1 Residual computation and attention

Let $R^{(\ell)} \in \mathbb{R}^{T \times d}$ denote the residual stream before transformer block ℓ , where T is the sequence length and d is the residual width. For a pre-normalisation decoder-only transformer, one block can be written as

$$R^{(\ell+\frac{1}{2})} = R^{(\ell)} + \sum_{h=1}^H A_h^{(\ell)} \left(N_{\text{attn}}^{(\ell)}(R^{(\ell)}) \right), \quad R^{(\ell+1)} = R^{(\ell+\frac{1}{2})} + M^{(\ell)} \left(N_{\text{mlp}}^{(\ell)}(R^{(\ell+\frac{1}{2})}) \right).$$

Here N denotes the model’s LayerNorm or RMSNorm operation. For a normalised input X , attention head h has the form

$$A_h^{(\ell)}(X) = \text{softmax} \left(\frac{(XW_Q^{(\ell,h)})(XW_K^{(\ell,h)})^\top}{\sqrt{d_h}} + \mathcal{M} \right) XW_V^{(\ell,h)}W_O^{(\ell,h)},$$

where $\mathcal{M}$ is the causal mask (Vaswani et al., 2017). The attention weights determine which earlier positions are read, while $W_V^{(\ell,h)}W_O^{(\ell,h)}$ determines what the head writes to the residual stream. An attention pattern can therefore look relevant even when its residual contribution has little effect on the answer.

The additive residual stream also permits local logit bookkeeping if the final normalisation is frozen at the activation being analysed. If N_t is that frozen linear map and $r_{t,c}$ is residual contribution c at position t , then

$$z_{t,y} = b_y + \sum_{c \in \mathcal{C}} u_y^\top N_t r_{t,c}.$$

This decomposition is not globally exact for LayerNorm or RMSNorm, and it is not a causal decomposition. Later attention, nonlinearities and normalisation can change the effect of a component (Elhage et al., 2021).

2.2 Patching and automated circuit discovery

Activation patching (Vig et al., 2020; Meng et al., 2022) compares a clean input x^{clean} with a corrupted input x^{corr} . At site s , the clean activation is inserted into the corrupted run and its effect on a task metric m is measured:

$$a_s^{\text{patch}} \leftarrow a_s(x^{\text{clean}}), \quad \Delta_s = m\left(f(x^{\text{corr}}; a_s \leftarrow a_s^{\text{patch}})\right) - m(f(x^{\text{corr}})).$$

The metric may be answer accuracy, logit difference, loss, or a calibrated probability; each choice answers a different question. A patch that restores a single logit may leave the remaining distribution changed, and a result on one prompt does not establish a reusable mechanism; verifying a circuit hypothesis requires counterfactual resampling under a declared correspondence (Chan et al., 2022).

Path patching isolates the effect transmitted from a sender through selected receiver paths while holding other routes to their corrupted values (Goldowsky-Dill et al., 2023). It is closer to the edge-level language of circuits than ordinary node patching, but its conclusion remains intervention-specific.

Automated Circuit Discovery (ACDC) recursively tests whether candidate edges can be pruned while preserving a metric (Conmy et al., 2023). On GPT-2 Small’s greater-than task, ACDC recovered all five previously described component types while retaining 68 of roughly 32,000 candidate edges. The experiment does not test whether the retained graph is unique or sufficient outside that prompt family.

Edge Attribution Patching (EAP) linearises the effect of replacing an edge activation by using a gradient at the corrupted run (Syed et al., 2024):

$$\text{EAP}(e) = \nabla_{a_e} m(f(x^{\text{corr}}))^{\top} \left(a_e(x^{\text{clean}}) - a_e(x^{\text{corr}}) \right).$$

Because one clean forward pass, one corrupted forward pass, and one backward pass score many edges, EAP is dramatically cheaper than exhaustive patching. The price is the first-order approximation. Saturated nonlinearities can have small local gradients despite large finite effects. EAP with integrated gradients (EAP-IG) evaluates the gradient along an interpolation path and improves faithfulness on several tasks, but still sometimes trails circuits selected by exact activation patching (Hanna et al., 2024). EAP with a gradient-path construction reports further gains on six tasks at higher cost (Zhang et al., 2025).

In this protocol, attribution is used only to rank hypotheses. Nanda’s attribution-patching note describes the method as exploratory and warns that the linear approximation is poor for large residual-stream replacements (Nanda, 2023). Every candidate retained after discovery is therefore checked by exact replay and tested outside the discovery split.

2.3 Superposition as overcomplete representation

Suppose a residual vector $x \in \mathbb{R}^d$ encodes latent features f_i using an overcomplete dictionary:

$$x = Df + \varepsilon = \sum_{i=1}^m f_i d_i + \varepsilon, \quad D = [d_1, \dots, d_m] \in \mathbb{R}^{d \times m}, \quad m > d.$$

When the features are sparse across examples, the model can represent more features than orthogonal dimensions by allowing $\langle d_i, d_j \rangle \neq 0$. Toy models show how feature importance and

sparsity affect this representation (Elhage et al., 2022). A sparse autoencoder (SAE) estimates a code z and reconstruction $\hat{x}$:

$$z = \sigma(W_{\text{enc}}(x - b_{\text{dec}}) + b_{\text{enc}}), \quad \hat{x} = W_{\text{dec}}z + b_{\text{dec}},$$

with a reconstruction and sparsity objective such as

$$\mathcal{L}_{\text{SAE}} = \mathbb{E}_x \left[\|x - \hat{x}\|_2^2 + \lambda \|z\|_1 \right].$$

Thresholded JumpReLU objectives are another option (Rajamanoharan et al., 2024). SAEs have produced interpretable features in small and large language models (Cunningham et al., 2023; Bricken et al., 2023; Templeton et al., 2024; Gao et al., 2024), but the learned dictionary is not known to be unique.

Three limitations of SAE-based circuit claims are relevant here: supervised distributed alignment search (DAS) can outperform SAE features for controlling known causal variables (Geiger et al., 2024; Mueller et al., 2025); SAEs can split, merge or omit causal variables, and on indirect object identification (IOI; Wang et al., 2023) supervised features control better than standard SAEs, though Gated and Top-K variants are competitive (Makelov et al., 2025); and dictionary size changes the apparent units (Leask et al., 2025). Section 5.4 reviews this evidence in detail.

Superposition can make an edit poorly conditioned. A causal variable may still occupy a recoverable subspace even when no single neuron represents it. Its stability, position on the activation manifold and downstream reuse must be tested.

2.4 Chain of thought (CoT): scratchpad, control signal, and report

Chain-of-thought prompting improves performance on many multi-step tasks, especially at sufficient scale (Wei et al., 2022), and zero-shot prompts such as “Let’s think step by step” can elicit similar behaviour (Kojima et al., 2022). Scratchpad training makes an important computational resource explicit: generated tokens expand the effective recurrent state available to an autoregressive model (Nye et al., 2021). The token sequence can function as external memory because later positions attend to earlier intermediate results.

Generated rationales are not automatically faithful reports of the computation. Turpin et al. (2023) found that answer-changing biasing cues could be omitted from explanations. Lanham et al. (2023) used truncation, mistake insertion, paraphrasing and related interventions, and found substantial variation across tasks and models. Tests using six classes of answer hints likewise found low disclosure rates for several reasoning models, with strong variation by model and hint (Y. Chen et al., 2025).

Related work separates visible deliberation from internal computation. Filler and pause tokens can carry consequential computation that never surfaces as an interpretable step (Pfau et al., 2024; Goyal et al., 2024; Baherwani et al., 2026), an answer can be decodable from hidden activations well before the trace verbalises it (Boppana et al., 2026), and an internal error signal can be present without conferring causal leverage on downstream behaviour (Yuan et al., 2026). Causal interventions also show that activations encode which entity is bound to which attribute through binding-ID vectors, information the text leaves implicit (Feng & Steinhart, 2024). These studies establish that consequential computation can occur without an interpretable visible trace; they do not identify a specific semantic state or test whether replacing that state causes the corresponding counterfactual transition.

The visible tokens can still matter even when the rationale is incomplete. Four roles are useful for planning interventions:

Table 1: *Four causal roles that can coexist within one chain of thought.*

Role	Causal function	Diagnostic intervention
External memory	Stores a partial result used later	Replace the value token; downstream computation should update coherently
Control anchor	Changes search phase or allocates computation	Remove or substitute discourse markers; inspect later routing and accuracy
Lossy report	Correlates with hidden state but omits determinants	Decode or patch hidden state while holding text fixed
Rationalisation	Is generated after the answer-causing state is fixed	Perturb the rationale without changing the answer; patch earlier decision state

A generated trace can contain several of these roles. A numeral may provide external memory, a discourse marker may affect routing, and a later sentence may summarise an earlier decision. Their causal contributions therefore have to be tested separately.

3. Counterfactual state authority

3.1 Why behaviour alone cannot settle the question

Consider a task with semantic state s_t , input or action a_t , and deterministic transition

$$s_{t+1} = F(s_t, a_t).$$

For column addition, s_t contains the digit index, carry and partial output. Let the model’s corresponding low-level state be h_t , with transition $h_{t+1} = G(h_t, a_t)$ and an alignment $\alpha : \mathcal{H} \rightarrow \mathcal{S}$ from model states to semantic states. The experiment seeks a part of h_t that can be set to represent a valid alternative state.

Behavioural underdetermination. A finite collection of prompt, trajectory and answer triples cannot identify an algorithm by itself. A lookup table indexed by the observed prompts and prefixes can return the same next-token distributions as a mechanism that computes the task. The two mechanisms differ only on counterfactuals outside the observed collection. More examples may improve coverage, but interventions are needed to distinguish the proposed mechanisms.

3.2 Interchange interventions and the commuting test

Let P_U project a model activation onto candidate subspace U , and let $r(s') \in U$ represent counterfactual semantic state s' . A state-only interchange intervention is

$$I_{U,s'}(h) = (I - P_U)h + r(s').$$

In practice, $r(s')$ may be taken from a compatible donor activation or estimated with a supervised subspace method. Interchange interventions provide a way to test a proposed causal abstraction (Geiger et al., 2021, 2022, 2025). I require donor matching and random controls because a raw donor activation may carry surface information in addition to the intended state (Figure 1).

Counterfactual transition fidelity (CTF). On an evaluation distribution $\mathcal{D}$ over low-level contexts h , actions a and valid counterfactual states s' , define

$$\text{CTF}(U, r; \mathcal{D}) = \mathbb{E}_{(h, a, s') \sim \mathcal{D}} [\kappa(\alpha(G(I_{U, s'}(h), a)), F(s', a))],$$

where κ is exact state agreement or a declared similarity score. I set a stricter primary endpoint: $\kappa = 1$ only when every downstream transition and the final answer match the executable counterfactual oracle, i.e. the per-transition score specialised to the full-trace indicator of Section 4.5. CTF is reported against same-norm random, shuffled-donor and unrelated-donor controls, and separately for each data split.

Nuisance invariance gap (NIG). Let τ change the renderer while preserving the semantic program and intervention. The pushed-forward paired distribution is $\tau_{\#}\mathcal{D}$, and

$$\text{NIG}(U, r; \mathcal{D}, \tau) = \text{CTF}(U, r; \mathcal{D}) - \text{CTF}(U, r; \tau_{\#}\mathcal{D}).$$

A small paired NIG indicates that the intervention result is stable across the two renderings. The value is signed and is reported with its interval; it is not replaced by an absolute value. The executable NIG conditions on an exact untreated trace under both renderings.

Operational criterion on a tested support. For a declared set of transitions $\mathcal{T}$, the model realises the tested causal abstraction when, for every $(s', a, F(s', a)) \in \mathcal{T}$ and every compatible nuisance realisation h ,

$$\alpha(G(I_{U, s'}(h), a)) = F(s', a).$$

The criterion is stated universally, but the executable test estimates the corresponding rate under $\mathcal{D}$ with a confidence bound; a guard “excludes” its cut-off when the bound on the adverse side clears it. The alignment α is realised here by decode-and-parse of the generated suffix rather than by a learned map on hidden states. This tests one transition only; a multi-step rollout would need a separate closure or homomorphism assumption. I therefore test the decoded suffix directly with full-trace CTF. Length-ODD and carry-pattern-ODD data enlarge only the support on which that direct test has been carried out.

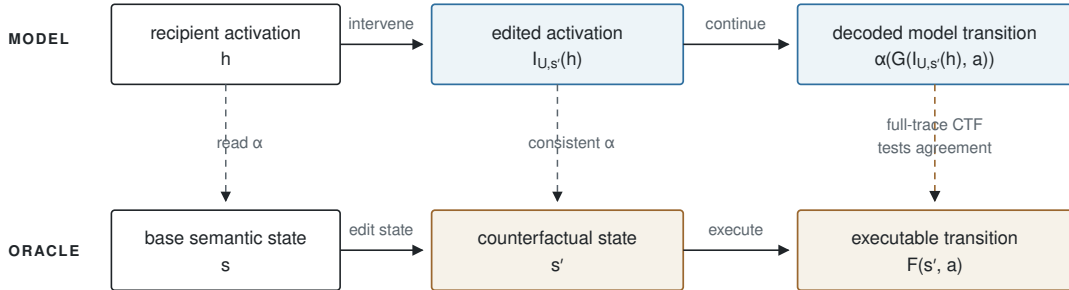


Figure 1: Interchange test for counterfactual state authority. Success requires the edited model transition to agree with the executable semantic transition.

3.3 Coordinate dependence

Mechanistic claims are sometimes attached to a raw coordinate, such as a particular neuron. Such labels are useful for locating an implementation, but the coordinate need not be unique.

Coordinate dependence at a linear interface. Suppose $h = W_{\text{in}}x$ is passed directly to $y = W_{\text{out}}h$, with no normalisation or nonlinearity at this interface. For any $A \in \text{GL}(d)$, set

$$h' = Ah, \quad W'_{\text{in}} = AW_{\text{in}}, \quad W'_{\text{out}} = W_{\text{out}}A^{-1}.$$

Then $W'_{\text{out}}h' = W_{\text{out}}h$. This calculation shows coordinate non-uniqueness at a linear bottleneck. It does **not** permit an arbitrary basis change through a complete transformer: in general, $\text{LN}(Ah) \neq A \text{LN}(h)$ and $\text{RMSNorm}(Ah) \neq A \text{RMSNorm}(h)$. Residual addition, learned element-wise gains, attention dot products and nonlinearities impose further restrictions. Permuting MLP neurons or attention heads together with their associated weights remains a genuine symmetry. I therefore specify the causal object by its intervention and measured effect; a coordinate name alone does not fix it.

Park et al. (2024) sharpen this warning for linear representations. They formalise representation directions through counterfactuals and derive a generally non-Euclidean causal inner product; geometric conclusions can therefore depend on the inner product used to define projection and similarity. Their result does not validate residual patching, but it is a reason not to equate an arbitrary Euclidean direction with the semantic state before intervention evidence is available.

3.4 What superposition predicts about edit interference

For a local calculation, assume unit-norm dictionary columns d_i and tied linear readouts $r_j(x) = d_j^\top x$. Edit feature i by setting $x' = x + \Delta d_i$. The intended readout changes by Δ , while an off-target readout changes by

$$\Delta r_j = d_j^\top (\Delta d_i) = \Delta \langle d_j, d_i \rangle, \quad j \neq i.$$

Consequently,

$$\sum_{j \neq i} (\Delta r_j)^2 = \Delta^2 \sum_{j \neq i} \langle d_j, d_i \rangle^2 \leq (m-1) \mu^2 \Delta^2, \quad \mu = \max_{i \neq j} |\langle d_i, d_j \rangle|.$$

The bound above is a worst case. For the average case, suppose the columns were drawn independently and uniformly from the unit sphere; then $\mathbb{E}[d_j d_j^\top] = I/d$ and $\mathbb{E} \langle d_i, d_j \rangle^2 = 1/d$ for $j \neq i$, giving

$$\mathbb{E} \left[\sum_{j \neq i} (\Delta r_j)^2 \right] = \Delta^2 \frac{m-1}{d}.$$

Total interference then grows linearly with the overcompleteness ratio m/d , however small μ becomes. Dictionary design cannot escape this scaling: the Welch bound gives $\mu^2 \geq (m-d)/(d(m-1))$, which approaches $1/d$ as m grows (Welch, 1974). Learned dictionaries are not random, but the reference case shows that a low-coherence dictionary can still carry large total interference, spread thinly across many readouts.

For an active set of size k , $k-1$ replaces $m-1$. For an untied downstream readout w_j , the off-target term is $\Delta w_j^\top d_i$. This local bound does not account for model nonlinearities, normalisation,

the activation manifold or repair by redundant paths. It shows why reconstruction loss and sparsity alone are insufficient measures of edit selectivity.

3.5 Information bottlenecks and visible rationales

The information-bottleneck objective seeks a representation Z that compresses input X while preserving information about target Y (Tishby et al., 1999):

$$\min_{p(z|x)} I(X; Z) - \beta I(Z; Y).$$

Let R be the visible rationale prefix and H the hidden state after that prefix. An informational diagnostic is the rationale sufficiency gap

$$\Delta_{\text{suff}} = I(Y; H \mid R).$$

A large value indicates that the hidden state contains answer-relevant information absent from the visible prefix. Mutual information by itself does not establish that the model uses this information. The causal test I use intervenes on R and H separately. For a clean and corrupted pair, a normalised patching-recovery score is

$$\text{Recovery}(M) = \frac{m(f(x^{\text{corr}}; M \leftarrow M(x^{\text{clean}}))) - m(f(x^{\text{corr}}))}{m(f(x^{\text{clean}})) - m(f(x^{\text{corr}}))}, \quad M \in \{R, H, (R, H)\}.$$

Values above one or below zero are possible because the channels can interact. They should be reported without clipping. The score is undefined when the corruption has no measured effect on the metric; as with the repair index in Section 4.5, denominators below 10^{-6} in magnitude are recorded as undefined. These two diagnostics motivate the factorial design; version 1 does not report them as endpoints.

3.6 Evaluation dimensions

For a claimed mechanism M , the following dimensions should be reported separately:

Table 2: *Evaluation dimensions for a candidate mechanism. Mediation and stability are listed for completeness; the version-1 gate covers transition fidelity, nuisance invariance and compositional reach.*

Axis	Question	Minimum evidence
Mediation	Does causally transmit answer-relevant information?	Exact clean/corrupt patches and multiple ablation baselines
Transition fidelity	Does a semantic state edit produce the correct next state?	Interchange interventions with matched donors
Nuisance invariance	Does the mechanism survive paraphrase and formatting changes?	Low NIG on held-out transformations
Compositional reach	Does it work on unseen state combinations and lengths?	Structured out-of-distribution split
Selective editability	Can the state be changed without broad damage?	Target success plus collateral suite
Stability	Is the description reproducible across seeds, examples, and discovery methods?	Cross-seed/cross-method overlap at the level of causal effect

These dimensions answer different questions. CTF and NIG test whether an edited state supports the expected transition across contexts; mediation estimates contribution to the answer; and collateral measures indicate whether the intervention is selective. Results are reported separately by model, task and evaluation split.

4. Proposed experimental protocol

The protocol specified here is centred on multi-digit addition and residual-stream interventions in open-weight decoder-only models.

4.1 Executable arithmetic traces

A semantic program is an operand pair together with its deterministic schoolbook execution. The addition generator samples two equal-width, unsigned base-10 operands. It executes schoolbook addition from the least significant column and records every transition. After zero-indexed column t , the semantic state is

$$s_t = (t, c_{t+1}, p_t),$$

where c_{t+1} is the carry into the next column and p_t is the partial result in least-significant-digit-first order. Each record also contains the operand digits, incoming carry, column total, output digit, outgoing carry, next transition and final answer. The counterfactual oracle I define changes one valid state field and recomputes the complete suffix with the same interpreter. A record is one JSON object per line (JSONL); for example, the first transition of $47 + 58$ is

```
{
  "column": 0, "operand_digits": [7, 8], "carry_in": 0,
  "column_total": 15, "output_digit": 5, "carry_out": 1,
  "next_state": {"t": 0, "carry": 1, "partial": 5},
  "final_answer": 105}
```

The canonical renderer prints the state fields in a fixed key order; the surface renderer changes the opening wording and instruction-line layout while preserving the schema. Generation is

greedy. A malformed, truncated or reordered generated trace is invalid; an executed invalid trace receives CTF zero.

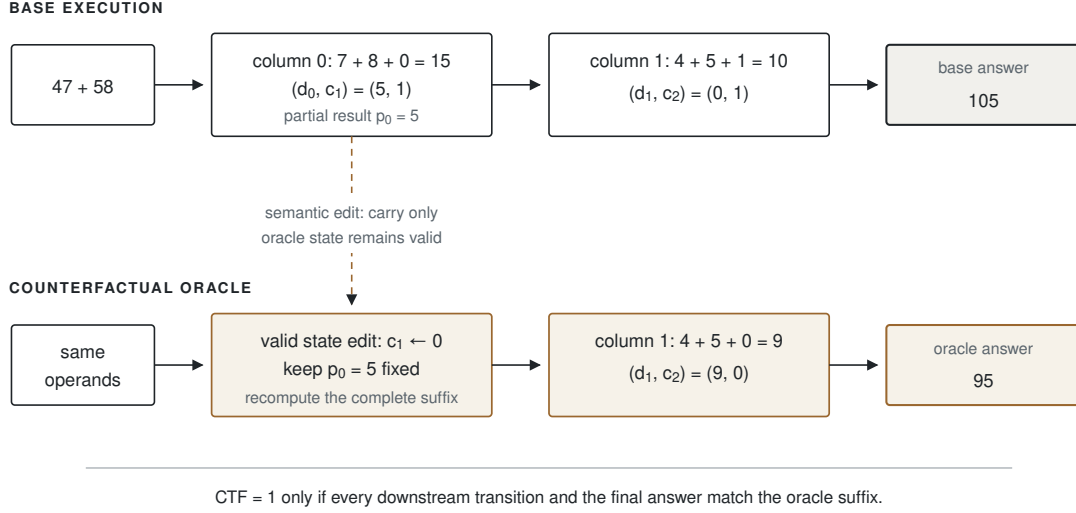


Figure 2: Executable base and counterfactual traces for $47 + 58$. The intervention changes the carry from 1 to 0 while keeping the first partial-result digit fixed at 5; full-trace CTF requires the complete suffix and final answer to match the oracle. The diagram specifies the test and contains no model output.

I evaluate addition on four distributions:

Table 3: *Addition splits used in the primary study.*

Split	Held out	Purpose
IID	Random semantic programs	Baseline reliability
Surface OOD	A second renderer for the same program	Sensitivity to wording and layout
Carry-pattern OOD	Complete carry-out signatures	Generalisation to unseen transitions
Length OOD	Five- and six-digit operands	Generalisation beyond support lengths

I fixed the primary configuration to **Qwen/Qwen2.5-1.5B-Instruct** in bfloat16, prospectively, because it is an open-weight instruction-tuned decoder with direct residual-stream access, is small enough for exhaustive layer screening on suitable hardware, and has independent arithmetic-mechanism evidence on the same model family: Bertolazzi et al. (2025) report digit-alignment and error-detection heads in Qwen2.5-1.5B. The Qwen2.5 technical report also documents mathematics and instruction-following evaluations across the model family (Yang et al., 2025). However, neither source shows that this checkpoint can generate the exact JSONL arithmetic traces required here. In fact, it is a study-specific eligibility condition rather than an assumption: each required renderer/role stratum within a split must reach an unpatched full-trace fidelity point estimate of at least 0.95 before its intervention results can enter the validation or final decision gates of Section 4.6.

No baseline or intervention run has been completed under this configuration. At the start of any run, the Hugging Face Hub alias is resolved to immutable model and tokenizer commits; those

hashes, rather than `main`, are written to the run manifest. Optional replication presets for other checkpoints are kept separate from the frozen study configuration.

Table 4: *Semantic-program counts for the addition study under protocol version 3; the decision rules in Section 4.6 are the version-1 claim set. Each surface-OOD program has canonical and surface renderings.*

Stage	Split	Target programs	Donor programs
Discovery	IID	512	512
Calibration	IID	256	256
Validation	IID	512	512
Validation	Surface OOD	512	512
Final	IID	512	512
Final	Surface OOD	512	512
Final	Carry-pattern OOD	512	512
Final	Length OOD	512	512

These counts are per semantic split; state-level intervention cases are generated from every eligible non-final column. Every 2-to-4-digit split uses fixed 10%, 40% and 50% width strata; length OOD divides evenly between widths 5 and 6. At each supported width, carry-pattern OOD holds out the first canonically hash-ranked carry signature and its bitwise complement, ensuring both carry values remain donor-matchable at every position. The validation and final allocations are fixed at 512 programs per semantic split, a nominal 128-cluster margin over the decision-rule floor in Section 4.6. Discovery and calibration counts remain operational capacity choices.

Dataset generation requires a clean Git commit and records it with the purpose, seed and exact hashes. An existing dataset is verified and returned only when its content is identical; corrupt or different content is not overwritten. A prompt or protocol change therefore requires a new study identifier and output root. Each stage writes the exact planned trial count before inference. If the required compute or trace storage is unavailable, the configuration must be revised and re-hashed before opening model outcomes; completed cells will not be silently downsampled.

I use program hashes, rather than prompt text alone, for leakage checks (Table 4). Target and donor pools are disjoint. A compatible donor must match research stage, split, renderer, semantic position, covering token-span length, activation shape and declared nuisance fields, while differing in the target state. Covering BPE tokens may include invariant JSON delimiters; donor and receiver absolute token positions need not match for donor identification, but an executed transfer must still satisfy the equal-index eligibility rule of Section 4.4. I store cases without a compatible donor as `missing_donor`.

4.2 Separation of research stages

I keep discovery, calibration, validation and final evaluation in separate, hash-verified files. Discovery data rank layers and timing boundaries. Calibration data are used for exact replay and site selection. Validation data test the frozen candidate on held-out IID and surface pairs. Only candidates fixed after validation are evaluated on final IID, surface-OOD, carry-pattern-OOD and length-OOD data. The staging follows registered-reports logic: the analysis is fixed before outcomes are observed (Nosek et al., 2018).

The final files are created on disk with the other partitions, and the pipeline keeps the stages separate by loading only the relevant partition: discovery, calibration and validation do not read or deserialize final records. This is a pipeline access rule, not investigator blinding (Figure 3).

In each run manifest I record the model and tokenizer identifiers and immutable revisions, purpose, seed, dataset hash, Git commit, package lock, device and precision, decoding settings, layer hook and token boundary. Executed trials retain every generated token ID and decoded output. Authority full-vocabulary diagnostics are computed online and retain only their scalar results; autoregressive and teacher-forced logit vectors are discarded after use.

The hashed dataset contains the exact prompt bytes for each record. Pre-execution failures leave those artifacts null and carry an explicit status and reason, and missing experiments are never replaced or imputed. Pilot and smoke summaries are written under **diagnostics/** and carry **publication_evidence=false**; only the publication configuration can create new **publication/** output. Retained diagnostic artifacts made before this guard may still use that historical directory name, but they remain pilot or smoke evidence.

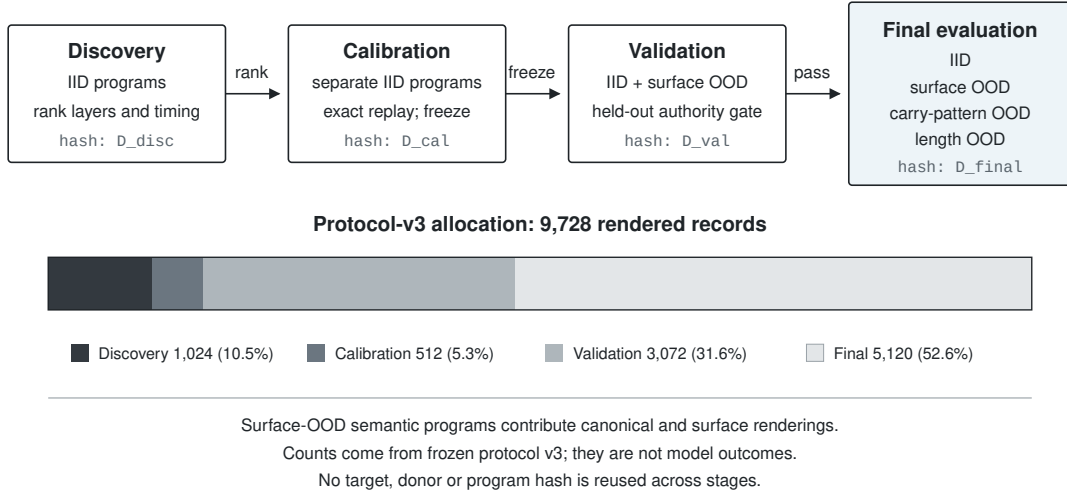


Figure 3: Data separation, candidate freezing and rendered-record accounting. Final files exist on disk, but discovery, calibration and validation do not load their records. The lower bar reports the protocol-v3 allocation; it is not model evidence.

4.3 Discovery and exact calibration

For each layer and timing boundary, discovery computes the first-order score

$$S_{\text{attr}}(s) = \nabla_{h_s} g^\top (h_s^{\text{donor}} - h_s^{\text{base}}),$$

Here g , with the gradient evaluated at the base run, is the counterfactual-token logit minus the base-token logit at the first aligned downstream token where the two oracle suffixes differ. It scores the first diverging token only; the complete transition is tested by exact replay. This score is used only for ranking. I then repeat each finite candidate in calibration with an exact replay and measure the change in the same logit gap.

The selection rule I apply groups calibration trials by task family, target field, timing and hook. Every program contributes one mean exact-replay effect and one mean same-norm random effect. In publication mode, a site is eligible only with at least 128 completed paired programs in both exact-replay and same-norm random arms, positive discovery attribution, positive exact-replay mean and exact-replay mean above the random-control mean. Smoke and pilot modes use a one-program floor so the path remains executable. For each family, field and timing, the eligible hook with the largest exact-replay mean is selected. An exact tie goes to the lower layer.

Calibration remains a ranking screen: it does not require all planned trials to finish or confidence-bound separation. The site table records program-level planned, exact-valid, random-valid and minimum counts; all accompany any selected-site result. A candidate that exists only as an attribution score may remain in the discovery audit table, where it is marked non-publishable, but it cannot enter validation or count as mechanistic evidence.

The executable study is limited to whole-block residual replacements. Its hidden-arm question is whether a compatible complete donor vector operationally steers the suffix at a selected boundary; success does not identify the named semantic state as the transported cause or localise it to a minimal head, neuron or direction. A compatible donor vector preserves covariance among features present in that activation and permits exact replay without fitting a second representation model on the calibration set. This makes the replacement a deliberately broad interchange test.

The breadth is also its main limitation. A successful replacement would show that the activation at the selected site controls the tested transition under this intervention. It would not show that the whole block computes the transition, that the semantic tuple is the only transferred information, or that a smaller component would suffice. ACDC, DAS, SAE discovery, head-level path patching and persistent weight editing remain follow-up analyses rather than evidence supplied by this study.

4.4 Intervention conditions and timing

The factorial design has four cells:

Table 5: *Hidden and visible intervention cells.*

Cell	Hidden residual	Visible state tokens
Y_{00}	Base	Base
Y_{10}	Exact compatible donor	Base
Y_{01}	Base	Counterfactual
Y_{11}	Exact compatible donor	Counterfactual

The hidden intervention replaces the complete residual activation at the selected site and is applied at three semantic boundaries: the prefix position that predicts the state value, the final emitted token of that field, and the delimiter immediately before the next operation. The visible intervention does not depend on the boundary. It forces the complete counterfactual `next_state` span of the edited record as one block while preserving the treatment-appropriate prefix; fields of that record printed before the span keep their base values, and every later field, including the next record’s incoming carry, is generated by the model. The forced record can therefore be visibly inconsistent with the operands, and a Y_{01} outcome is read against that self-contradictory context rather than as evidence about a consistent edit. Y_{01} is evaluated once per case and shared across boundaries, whereas Y_{10} and Y_{11} are evaluated at each boundary. Scoring begins at the first token the model generates after the intervention. A hidden patch at the first boundary lands before the state field is emitted, so the scored span there includes the field itself; a pass at that boundary is reported together with the fraction of cases whose emitted field changed. Base arms follow the same decoding procedure as treated arms; only the intervention content differs. The historical hidden patch is reapplied during each subsequent forward pass; generation does not rely on a stale key-value cache (Figure 4).

Subword token alignment for visible edits

The visible intervention is defined at the token level, and Qwen2.5 uses byte-level byte-pair encoding (BPE). A character span marking a state field can therefore cross a tokenizer boundary, and two values can occupy different numbers or positions of tokens. Each field is tokenised with the frozen run tokenizer before any model outcome is opened. Eligibility requires both character boundaries to coincide exactly with tokenizer offsets and the base, donor and counterfactual spans to occupy the same token indices, intervention boundary and activation position. An eligible visible edit therefore changes the forced token IDs without shifting downstream position indices. This removes the position-shift confound for eligible cases; it does not hold downstream hidden states fixed, because reading a different token is the intended treatment.

Misaligned fields are not repaired by splicing partial tokens or padding the shorter value. They are recorded as `token_misaligned` in the coverage ledger and excluded before treatment. I report them by split and target field, beside the coverage counts (planned, eligible, executed, metric-valid). This narrows inference to equal-position tokenisations. It applies to hidden donor transfer as well as the visible arm, because copying a residual from a different absolute position would mix the state edit with positional information.

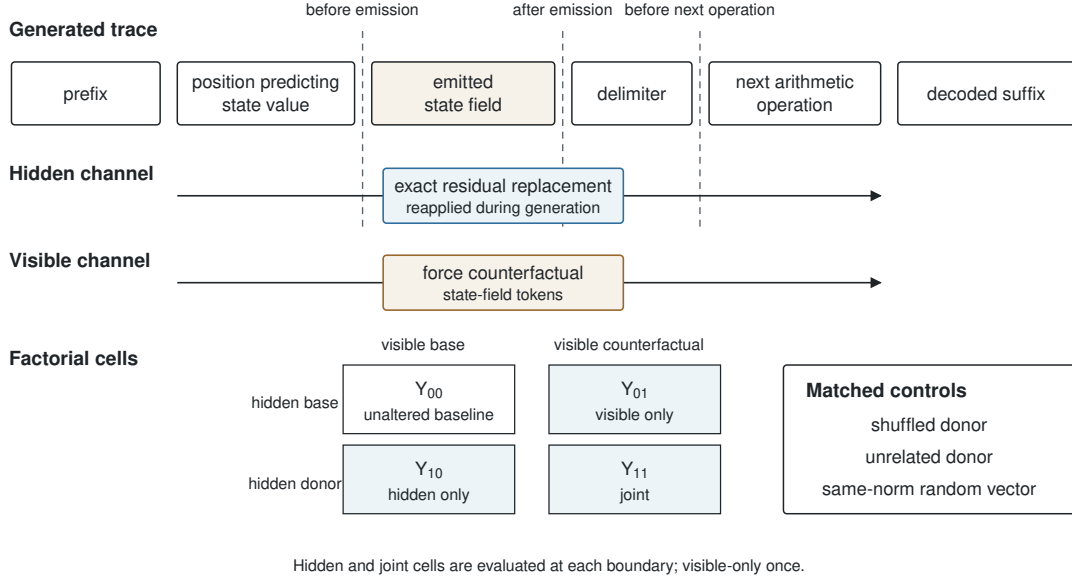


Figure 4: Hidden and visible intervention channels. Hidden patches are tested at the three semantic boundaries; the visible edit forces the state-field span once.

For the hidden and joint arms I use three controls at the same layer and timing. A shuffled donor has the required target state and comes from a second donor program that satisfies every compatibility rule of Section 4.1 except one: when a nuisance field is declared, it must differ from the active donor on at least one such field. Under the state hypothesis a second vector carrying the same state should reproduce the active result, so the shuffled donor is a replication check rather than a negative control: whether or not donor-identifying nuisance fields are declared, the simultaneous active-minus-shuffled CTF interval must lie within $[-0.10, 0.10]$. A result that depends on which donor supplied the vector therefore counts against clean state transfer; donor-specific carry-over is measured separately by donor leakage. An unrelated donor retains the base or an unrelated state. A seeded random vector is sampled on the sphere, scaled to the norm of $h^{\text{donor}} - h^{\text{recipient}}$ within the numerical precision of the activation dtype, and added to the recipient activation. The unrelated donor and the random vector are the negative controls.

Controls mirror the active channels of the intervention arm they test (Figure 5). I distribute each control across the compatible frozen pool with a deterministic case-to-donor hash rather than repeatedly taking its first record. These controls apply to the hidden and joint arms; the visible arm carries no donor content to leak and is guarded by the full-trace endpoint and the NIG alone. A forced-unrelated-token control is the natural analogue and is deferred rather than pre-specified.

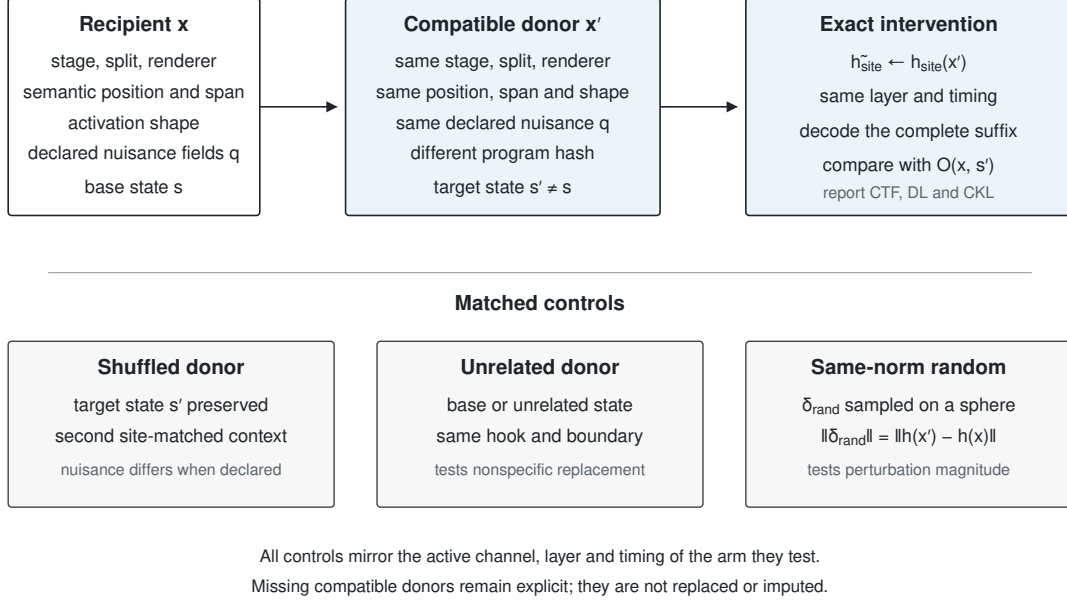


Figure 5: Compatible donor construction and matched controls. The active patch and each control use the same layer, timing and channel; they differ only in the vector written into that activation. Missing compatible donors remain missing.

4.5 Primary and diagnostic metrics

Let O_i be the executable counterfactual suffix and T_i the parsed model suffix. I take the primary CTF estimand to be conditional on an intact recipient baseline, an intact compatible-donor baseline where the cell uses one, and an executed intervention; inference is therefore over cases the unpatched model already solves. The executable estimand is program-weighted: each recipient program contributes the mean over its eligible cases, so programs with more eligible columns do not carry more weight, and this replaces the case-level expectation of Section 3.2 whenever programs contribute different numbers of cases. Pre-execution failures remain in the coverage ledger but do not enter the metric denominator. Once an intervention is executed, a malformed, truncated or otherwise invalid continuation receives CTF zero. Full-trace fidelity is

$$\text{CTF}_i = \mathbf{1} \left[\begin{array}{l} T_i \text{ is valid,} \\ \text{every downstream transition agrees with } O_i, \\ \text{the final answers agree} \end{array} \right].$$

The state-authority contrasts use cell means:

$$H = Y_{10} - Y_{00}, \quad V = Y_{01} - Y_{00}, \quad J = Y_{11} - Y_{00}, \quad I = Y_{11} - Y_{10} - Y_{01} + Y_{00}.$$

Here I denotes the interaction contrast, not the interchange operator $I_{U,s'}$ of Section 3.2.

These are intervention-response contrasts; a complete mediation decomposition would need a different design. A label of hidden, visible, both, neither or indeterminate is assigned only after the pre-specified CTF, control, NIG and leakage confidence bounds are applied.

Repair is measured from the model’s preference between complete counterfactual and base semantic fields. If $L_k(C)$ and $L_k(B)$ are their length-normalised log likelihoods at checkpoint k , then

$$S_k = \frac{\exp L_k(C)}{\exp L_k(C) + \exp L_k(B)}, \quad E_k = \frac{S_k^{\text{int}} - S_k^{00}}{1 - S_k^{00}}, \quad \text{RI} = 1 - \frac{\bar{E}_{\text{final}}}{\bar{E}_{\text{immediate}}}.$$

RI is not clipped. Zero indicates persistence of the immediate effect and one indicates full repair; negative values and values above one are retained. The index is descriptive; version 1 assigns it no pass or fail cut-off. Coincident candidates or denominators below 10^{-6} in magnitude are explicitly undefined.

For nuisance fields $\mathcal{F}_i$ whose donor value differs from both the base and the counterfactual oracle, donor leakage is

$$\text{DL}_i = \frac{1}{|\mathcal{F}_i|} \sum_{f \in \mathcal{F}_i} \mathbf{1}[T_{i,f} = D_{i,f}].$$

An invalid generated trace makes donor leakage undefined rather than evidence of no copying; an empty eligible field set is likewise undefined and does not create a leakage requirement. The current collateral diagnostic is the mean full-vocabulary KL divergence at aligned post-boundary token positions $\mathcal{P}_i$ where the base and counterfactual oracle token IDs are equal:

$$\text{CKL}_i = \frac{1}{|\mathcal{P}_i|} \sum_{p \in \mathcal{P}_i} D_{\text{KL}}(p_{00,p}(\cdot) \parallel p_{\text{int},p}(\cdot)).$$

I compute log-probabilities in float64, without smoothing, clipping or top- k truncation; positions where the intervened distribution assigns zero probability to a token the base distribution assigns positive mass are recorded as infinite rather than truncated. Note that equality of oracle token IDs is only a token-level alignment rule; CKL is not presented as a semantic-invariance or safety certificate. Every active cell also receives a damage report containing the repair index, donor leakage and field coverage, CKL with its interval, trace-validity rate, and the coverage counts. The report is descriptive: version 1 has no CKL rejection threshold, and a low mean CKL cannot rule out a concentrated failure at an unmeasured token or task.

4.6 Decision rules and reporting

Primary intervals come from one deterministic hierarchical bootstrap with 200{,}000 replicates, and every interval is read from the same replicates. Because one donor program may serve several recipients, each replicate resamples recipient programs and donor programs independently (a two-way cluster bootstrap); a case enters with the product of its recipient’s and its donor’s multiplicities, and visible-only cases, which have no donor, enter with the recipient multiplicity alone. State-level cases from the same recipient program share a cluster, and every sampled program contributes one equally weighted within-program mean. Canonical and surface rows are paired before the corresponding surface-program cluster is resampled. The intervals are percentile intervals. At the level used here each two-sided tail quantile rests on about 357 replicates

(about 714 for a one-sided bound), but percentile coverage at such small tail probabilities is only approximate near the cluster floor. Bonferroni-adjusted interval levels target nominal simultaneous 95% coverage over the evidence used to classify one frozen candidate site. One interval family is instantiated per evidence family. The paired-surface family contains 14 intervals: for each of Y_{10} and Y_{11} , the CTF bound, the advantages over the unrelated-donor and random controls, the shuffled-donor replication interval, the NIG interval and the donor-leakage bound; for Y_{01} , the CTF bound and the NIG interval. The IID family omits the three NIG intervals, which require paired renderings of the same program. The divisor stays at 14 in every family even where an interval does not apply, so the correction does not depend on realised data, and the central interval level is $1 - 0.05/14 = 0.99643$. Each family carries its own 0.05 budget, and every classification is stated per evidence family; the study makes no joint coverage claim across families. This coverage statement is site-specific; the study reports every frozen site and makes no omnibus claim that at least one site in the set is authoritative. I count an arm as supported only when the simultaneous lower CTF bound is at least 0.80, its lower paired advantages over the unrelated-donor and random controls are each at least 0.10, its shuffled-donor replication interval and its paired NIG interval each lie within $[-0.10, 0.10]$, and, when donor-identifying nuisance fields are declared, the upper donor-leakage bound is at most 0.05. The visible arm has only the CTF and NIG guards.

I fix a prospective floor of 384 independent complete semantic-program clusters for every applicable IID and paired-surface evidence family. A cluster is complete for a decision when it contributes at least one case in which every arm and control that the decision uses was executed and scored, with both renderings for paired-surface evidence. The floor applies to the set each decision actually uses: the per-arm set for advancing a site, and the common complete-case set of Section 6.2 for assigning a label. The floor is an operational capacity choice fixed before any model outcome is opened, not a quantity estimated from data; all estimates, intervals, completion counts and the damage report are still shown below it, but the arm cannot be supported or ruled out and the profile remains indeterminate. Repeated state rows from one program do not increase this count. Carry-pattern-OOD and length-OOD results are reported descriptively and carry no separate cluster floor. The 512-program validation allocation leaves a 25% margin, and that margin applies to the set the label uses. With k independent completion conditions each retained at rate q , joint completion is q^k , so keeping 75% of programs jointly needs $q \geq 0.75^{1/k}$: about 0.93 for four conditions and 0.98 for fifteen. If qualification runs indicate joint completion below 0.75, the allocation is enlarged and the configuration re-hashed before any intervention outcome is opened (Section 4.1); it is never topped up afterwards. I did not relax the interval correction because the same profile also rules an arm out when any one guard excludes its cut-off; a within-arm intersection-union argument protects support, but not false exclusion across several guards. The decision rule does not calibrate CKL and supplies no basis for a selectivity threshold.

The floor also fixes what the CTF guard can detect. In the least informative case, where cases within a program are perfectly correlated and each of 384 programs counts as one Bernoulli outcome, the lower bound at level 0.99643 reaches 0.80 only when observed CTF is at least about 0.853. The probability of clearing the CTF guard is then about 0.44 at a true CTF of 0.85, 0.84 at 0.87, 0.95 at 0.88 and above 0.99 at 0.90; weaker within-program correlation raises these figures, and requiring every other guard to pass lowers them. The design is therefore sensitive only to near-complete transfer: an arm with true fidelity near 0.85 will often be classified as indeterminate rather than supported.

So a candidate does not pass validation if exact replay fails, task-consistent effects disappear on held-out data, or applicable negative controls perform similarly. A split enters the authority gate only when the unpatched baseline full-trace fidelity point estimate on that split is at least 0.95; below this floor a patched fidelity near chance is uninformative, so the split is reported

as baseline-ineligible and can neither support nor refute authority. CKL is reported beside the authority result, but version 1 has no pre-specified CKL threshold and makes no selectivity or safety claim from it. I produce CSV and Parquet tables from immutable run rows and regenerate every figure deterministically from them (Figure 6). The coverage counts keep failed and missing runs visible without adding them to conditional metric denominators.

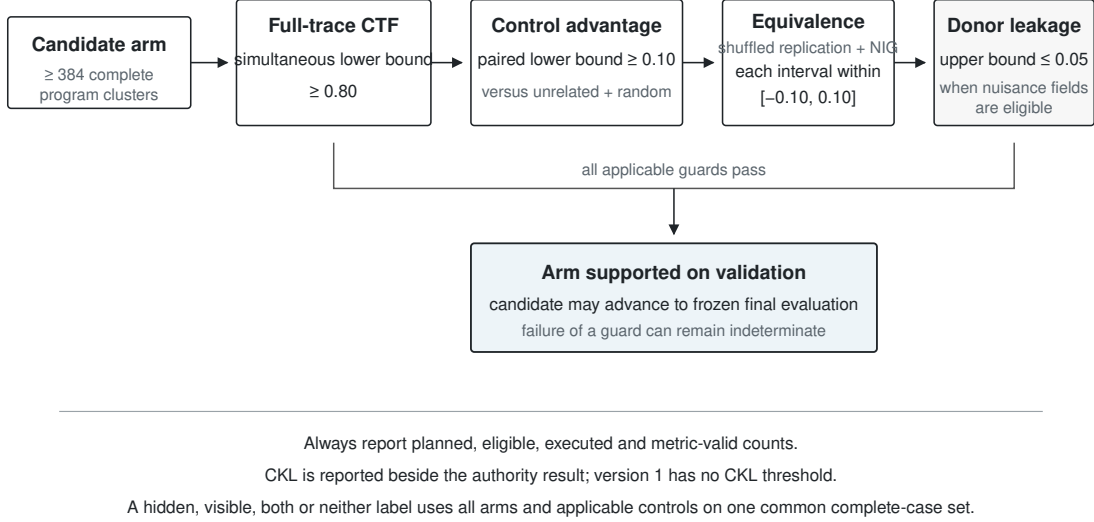


Figure 6: Pre-specified validation gate for one candidate arm. Support requires at least 384 independent complete program clusters and every applicable simultaneous confidence-bound guard. CKL and completion counts are reported but do not enter the version-1 gate.

5. Evidence from prior work

This section reviews studies that directly inform the protocol. Their models, tasks and intervention definitions differ, so the reported values should not be compared as if they came from one benchmark. Numerical details are attributed to the study from which they were taken.

5.1 Existence proofs: transformers can learn algorithms

The cleanest algorithmic evidence comes from tasks small enough to reverse engineer.

Modular addition. Nanda et al. (2023) analysed grokking in a transformer trained on $a + b \pmod{113}$. The learned computation embeds inputs using a small set of Fourier frequencies, combines them through trigonometric identities, and reads out the modular sum. Ablations and restricted loss curves track the emergence of this circuit. Zhong et al. (2023) later found qualitatively different clock-like and pizza-like mechanisms across training conditions, including models that used more than one mechanism. These results show that a model can generalise with a compact algorithm while the precise circuit still depends on training.

Iteration heads. Cabannes et al. (2024) constructed a two-layer transformer with one attention head per layer and studied synthetic chain-of-thought iteration. In their account, the first head locates the end of the input sequence, an MLP helps identify the current step, the second head retrieves the current item, and the last MLP and output map implement the successor operation. Pretraining transfers to parity, with successful fine-tuning in fewer than 30 epochs in their averaged 100-run study. This controlled model is relevant to the present transition test, though its scale and training setup differ from a pretrained language model.

Symbolic path finding. Brinkmann et al. (2024) trained a transformer on a symbolic multi-step reasoning task. “Deduction heads” climb a tree one level per layer while register-like tokens hold parallel subproblems. The learned strategy is depth-bounded: when the computation exceeds the available unrolling, the model falls back on heuristics. This result motivates reporting length OOD separately from IID transition fidelity.

Together, these studies show that transformer computations can be described at the level of intermediate transitions in controlled tasks. They do not estimate how often the same account applies to pretrained models. They also motivate two parts of this protocol: testing hidden and token positions separately, and keeping computation length as an OOD factor.

5.2 Narrow circuits in pretrained language models

GPT-2 greater-than

Hanna et al. (2023, Sections 3.2 and 3.3) reverse engineered GPT-2 Small on prompts such as “The war lasted from the year 17 32 to the year 17. . .”, where the model should prefer a two-digit completion greater than 32. Using zero-based indices, they identify seven upstream attention heads ($a9.h1$, $a8.h11$, $a8.h8$, $a7.h10$, $a6.h9$, $a5.h5$ and $a5.h1$) that transmit information about the comparison year to the prediction position. MLPs 9 and 10 directly raise logits for values above the comparison year; MLP 11 has a more restricted positive effect, and MLP 8 acts mainly through later components.

Targeted interventions and a hand-specified circuit recover much of the behaviour, and ACDC later rediscovered its component types (Conmy et al., 2023). The comparison is performed in a single forward pass over a rigid prompt family, so it does not address visible chain-of-thought.

Arithmetic as a bag of heuristics

Nikankin et al. (2025) analysed arithmetic in pretrained language models and found causally important middle-to-late MLP neurons associated with recognisable heuristics. The reported categories include magnitude and digit-pattern rules that combine to produce an answer; one reported family fires when an operand or result is a round number, while others track which digit occupies a given position. The study interprets the mechanism as a collection of heuristics instead of one general arithmetic routine.

The carry-pattern and length splits test whether these heuristics remain stable under counterfactual changes.

Quirke and Barez (2024) reverse engineered a one-layer transformer trained from scratch on digit-token addition. In the three-head configuration analysed most closely, time-offset heads form a double-staircase attention pattern over operand pairs. Their account decomposes middle output digits into a base addition signal and two carry-related signals from adjacent columns; the MLP combines them one token before output. Cascading carries through runs of nines remain a rare high-loss case. A small addition model can therefore distribute carry logic across parallel local computations rather than maintain one serial schoolbook tuple. Whether Qwen2.5 uses the same mechanism remains untested.

Using causal mediation on arithmetic questions, Stolfo et al. (2023) found that early and middle attention components transmitted query information to the final token, after which later MLPs supplied result-related information. The finding concerns information flow in the tested models; whether an exact schoolbook-addition state machine exists remains open.

Two further results bound what the present test can expect. Levy and Geva (2025) found per-digit circular base-10 representations in Llama 3 8B and Mistral 7B, and their digit-direction

edits affected generated number values. This motivates digit- and carry-specific hypotheses, but it does not establish that the models tested here encode the full addition state.

Li et al. (2024) prove an expressivity result: with T autoregressive intermediate steps and logarithmic embedding dimension, a constant-precision, constant-depth transformer can express functions computed by Boolean circuits of size T . Their experiments compare chain of thought, hints and direct prediction on modular addition, permutation composition, iterated squaring and circuit value, with the clearest chain-of-thought gains on the tasks treated as inherently serial. The result explains how emitted tokens can add sequential computation. Left open are whether those contents are faithful, whether ordinary multi-digit addition requires visible tokens, and whether an instruction-tuned Qwen model naturally relies on its printed carry.

Arithmetic validation heads

Bertolazzi et al. (2025) studied whether small language models can recognise that a displayed arithmetic result is inconsistent with a final answer. In Qwen2.5-1.5B, layer 12, head 2 aligns corresponding digits between the arithmetic result and answer across the study’s 5,000-prompt analysis (their Section 4.2). The authors distinguish lower- and middle-layer consistency heads from later computation components. Their interventions on six consistency heads control error detection, and a residual-stream feedback intervention improves detection of consistent but incorrect answers (their Sections 4.2 and 4.3).

The interventions identify a consistency component in a sizeable prompt set, but agreement checking and arithmetic recomputation remain distinct tasks.

Multi-step deduction in Llama-2

Dutta et al. (2024) analysed Llama-2 7B on PrOntoQA deduction. Heads implicated in decision, copying and induction (Olsson et al., 2022) overlap substantially. The paper reports multiple parallel answer pathways and a division between earlier processing and later answer construction.

The result is compatible with a distributed and redundant mechanism. Because pruning can leave alternative paths intact, the retained set should not be treated as a unique reasoning circuit.

5.3 Reasoning traces as control plane and external memory

Thought anchors

Bogdan et al. (2025, arXiv:2506.19143v4, Sections 2.1 and 4.1) studied R1-Distill-Qwen-14B on 20 challenging MATH problems. They selected one correct and one incorrect trace for each problem, giving 40 responses. The responses averaged 144.2 sentences and 4,208 tokens. They identify receiver heads whose attention narrows towards particular sentences; layer 36, head 6 is one example. Split-half reliability of head kurtosis was $r = .84$. Among the 16 heads with the highest kurtosis, the mean pairwise correlation of sentence-level vertical-attention scores was .56, compared with .35 across arbitrary heads. However, these correlations concern attention concentration; they do not measure causal sentence influence directly.

In a separate 32-problem ablation study, accuracy was 48.8% after ablating 256 receiver heads and 52.7% after a layer-matched non-receiver control (their Appendix L). With 512 heads ablated, the corresponding values were 27.7% and 37.3%. The authors suggest that the long traces permit error correction or compensation, which may explain why a separation appears only under broad ablation. This supports a distributed routing role for receiver heads in the tested traces.

Attribution graphs in Claude 3.5 Haiku

Anthropic’s 2025 circuit-tracing work replaces MLP computations with cross-layer transcoders and constructs attribution graphs for selected behaviours (Lindsey et al., 2025; Ameisen et al., 2025). One case follows the prompt “Fact: the capital of the state containing Dallas is” through a Dallas → Texas → Austin feature path. Replacing the Texas feature cluster with a California cluster changes the answer to Sacramento; analogous swaps produce Atlanta for Georgia, Victoria for British Columbia, Beijing for China, and Constantinople for the Byzantine Empire. These counterfactual substitutions provide evidence beyond the semantic feature labels.

The same work contrasts a calculation of $\sqrt{0.64} = 0.8$ with examples where the model produces plausible intermediate claims for a difficult cosine question or reasons backwards from a supplied answer. The cases illustrate that generated explanations can relate to internal computation in different ways.

The attribution graph is built over a learned replacement model. The authors report omitted query-key attribution, approximation error and unexplained “dark matter” contributions. The feature swaps therefore establish control in selected cases while leaving the completeness of the graph unresolved.

Latent chains of thought

Continuous-reasoning methods such as COCONUT and CODI replace some verbal reasoning tokens with latent states (Hao et al., 2024; Shen et al., 2025). They demonstrate that useful intermediate computation need not be rendered as natural language. This weakens any attempt to define reasoning by the presence of an English transcript.

Liang and Pan (2026) analyse small CODI-style models on controlled polynomial tasks. In their three-layer, two-head setting, two- and three-hop problems produce bridge states in the latent channel, while the final answer uses a copy-like route from the input with late fusion of intermediate information. At longer hop counts the representation often contains only partial late intermediates. Under prime-modulus evaluation, explicit chain of thought performs better than the latent method. As a controlled example it is informative; its three-layer scale and polynomial tasks remain far from multi-digit addition in a pretrained model.

5.4 Superposition tools: progress and negative results

Sparse feature circuits combine SAE-like features with edge attributions. Marks et al. (2025) show that sparse features can produce compact causal graphs on several language behaviours. Dunefsky et al. (2024) evaluate transcoders in models from 120M to 1.4B parameters and demonstrate feature-circuit analysis on GPT-2 Small. These methods provide a feature-level view of communication between components.

Several reported limitations are relevant to the present study:

- Makelov et al. (2025) find that SAEs trained on IOI activations recover relevant features; standard SAEs control the known causal variables less well than supervised features, whereas Gated and Top-K SAEs are competitive on disentanglement and control. Feature occlusion and over-splitting remain.
- Leask et al. (2025) show that SAE dictionaries are not canonical: larger dictionaries reveal refinements absent from smaller ones, and meta-SAEs decompose apparently atomic latents.
- In the Mechanistic Interpretability Benchmark, attribution and mask-optimisation methods perform best at circuit localisation, supervised DAS performs best at causal-variable

localisation, and SAE features do not generally outperform neurons on that causal track (Mueller et al., 2025).

- X. Chen et al. (2026), an SAE study of CoT on Pythia-70M and Pythia-2.8B, report that patches from CoT to no-CoT runs can improve correct-answer logits in the larger model but are unstable in the smaller one. Randomly sampled CoT features outperform top-K most-salient features in the reported 2.8B experiments. In one patch curve, the reported $\Delta \log P$ rises from about 1.2 to 4.3. These analyses use the GSM8K training split.

SAE reconstruction and interpretability scores do not establish causal control, so SAE features require the same intervention tests as other candidate representations.

5.5 Faithfulness, ablation, and repair

Circuit metrics inherit the assumptions of their interventions. Miller et al. (2024) systematically vary ablation choices and show that the apparent faithfulness of transformer circuits can change substantially. Mean ablation, zero ablation, resampling and corrupted activations produce different intervention distributions and answer different counterfactual questions; zero and noise ablations have a particular risk of creating out-of-distribution activations.

McGrath et al. (2023) demonstrate the “Hydra effect”: after one layer is ablated, later components can increase their contribution and partially repair the model’s output. A component can be causally active in the intact model but appear unnecessary after compensation. Reasoning traces make this especially costly: several heads may encode the same intermediate result, or rederive it from the visible tokens themselves.

Lieberum et al. (2023) identify localised components for some behaviours in Chinchilla-scale models, including a 70B model. Their semantic descriptions remain partial and can fail on broader randomised-label distributions.

Hase et al. (2023) compare knowledge-localisation methods with subsequent factual editing and find that causal-tracing scores do not reliably identify the layer where an edit works best. Although this result concerns factual editing, it shows that a site carrying information may be a downstream bottleneck or a redundant copy.

The studies reviewed here provide causal evidence in several controlled tasks and partial evidence for distributed state and routing in larger models. Because their metrics and interventions differ, they are used to motivate controls rather than to support a general chain-of-thought claim.

Two recent studies bound the present design from opposite sides. Shih et al. (2026) create an editable state register through trace supervision and show that rewriting its internal value makes the model apply the next operation to the edited state; their setting trains the state into place and scores a single transition, whereas the present test targets an untrained elicited trace and requires the complete counterfactual suffix. The same work reports that replacing an activation at one layer recovers under 2% of the through-layer patch effect, direct evidence that a single-site whole-vector transplant need not reproduce the natural configuration — one reason a pass here is read as operational sufficiency rather than identified state transport. Yuan et al. (2026) report the complementary negative result: a probe decodes trace correctness at 0.95 AUROC, yet steering, best-of-N, self-correction and activation patching all fail to change behaviour, so the signal is diagnostic rather than causal — the gap a passing hidden arm here must cross. Macar et al. (2026) further find that off-policy edits to a reasoning trace have small and unstable effects relative to counterfactual resampling, which motivates the matched-donor and equal-position restrictions adopted above. Two closer precedents each cover one arm. Zhu et al. (2025) edit intermediate values printed in the chain of thought for multi-digit multiplication and dynamic programming and find that later tokens and the final answer change accordingly; that is the visible-only arm, without an executable full-trace oracle, a hidden-arm comparison or

pre-specified gates. Mehrafarin et al. (2026) patch hidden states of individual chain-of-thought tokens into direct-answer runs on GSM8K and recover answer information concentrated in mid-to-late layers; their donor is the same question’s own trace rather than an edited semantic state, and their endpoint is answer accuracy rather than transition fidelity.

6. Primary carry intervention

6.1 Worked counterfactual

Consider the least-significant-digit-first execution of $47 + 58$:

$$\begin{aligned} 7 + 8 + 0 &= 15 \Rightarrow (d_0, c_1) = (5, 1), \\ 47 + 58 : \quad 4 + 5 + 1 &= 10 \Rightarrow (d_1, c_2) = (0, 1), \\ 0 + 0 + 1 &= 1 \Rightarrow 105. \end{aligned}$$

Immediately before the tens-column transition, the intervention changes the carry from $c_1 = 1$ to $c_1 = 0$ while retaining the partial result digit 5. The executable counterfactual suffix is

$$c_1 \leftarrow 0 : \quad 4 + 5 + 0 = 9 \Rightarrow (d_1, c_2) = (9, 0) \Rightarrow 95.$$

Figure 2 depicts this branch and its full-trace scoring requirement. The target is the complete trace ending in 95; a higher logit for token 95 alone does not count. A continuation that returns to 105 is counted as failure of CTF and may show repair of the original state. I generate the same test at every eligible column, in both carry directions, and for partial-result edits. Donors are matched by the rules in Section 4.1. The counterfactual is valid in the state space — some operand pair produces a zero carry at that column — but it is unreachable from the receiver’s own operands, since $7 + 8 = 15$ forces $c_1 = 1$ in the unedited program. That deliberate history-inconsistency is the intervention, and it is also what lets a repair process recompute the original carry (Section 8.2). That is the entire test.

6.2 Interpretation of the four cells

The factorial conditions support eight primary descriptions:

Table 6: *Primary state-authority interpretations. “Ruled out” means that an arm’s simultaneous bounds exclude at least one guard cut-off (Section 4.6); it does not by itself mean that the intervention had no effect, and each row is reported with the guard that decided it. “Congruent” means the paired intervention carries the same counterfactual state.*

Validated pattern	Interpretation within this experiment
Y_{10} and Y_{11} pass; Y_{01} is ruled out	Hidden intervention passes the selected rule; visible intervention does not
Y_{01} and Y_{11} pass; Y_{10} is ruled out	Visible intervention passes the selected rule; natural dependence is not established
Y_{10} , Y_{01} and Y_{11} all pass	Both interventions are separately sufficient; interaction is reported separately
Only Y_{11} passes; Y_{10} and Y_{01} are ruled out	Only the congruent joint intervention passes; necessity is not identified
Y_{10} passes; Y_{01} and Y_{11} are ruled out	Hidden steering works alone but congruent visible forcing removes efficacy; an antagonistic pattern
Y_{01} passes; Y_{10} and Y_{11} are ruled out	Visible forcing works alone but congruent hidden replacement removes efficacy; an antagonistic pattern
Y_{10} and Y_{01} pass; Y_{11} is ruled out	Each channel suffices alone; the congruent pair fails, a non-additive interaction reported separately
All active arms are ruled out	No arm passes the selected rule; efficacy and guard failures are separated

I report two classifications: a CTF-only efficacy label and a guarded-authority label that additionally requires control advantage, nuisance invariance and low applicable leakage. A guarded failure therefore cannot be read as absence of efficacy. If confidence bounds do not support or rule out the required arms, the corresponding result is indeterminate. Visible-only means that no residual is directly replaced. It does not mean that later hidden states remain fixed: once the forced token is read, its embedding changes every subsequent residual computation. A passing Y_{01} arm therefore establishes the efficacy of that token intervention, not that the unpatched model naturally depends on the printed field or that a hidden representation is redundant. Natural necessity would require a separate equal-span corruption or ablation of the model’s own emitted field; full mediation would require a different design. The same caution applies to a successful hidden-only arm: it does not prove that the visible channel has zero mediated effect.

Applicable donor leakage enters the authority gate. CKL, the damage report and final OOD performance are reported separately and limit claims about selectivity and generality.

In validation I first restrict the four factorial cells to shared complete cases, then apply the controls required for each arm. I advance a site to final evaluation if at least one arm passes all of its guards. Assigning a hidden, visible, both or neither label is stricter: all arms and applicable controls use one common complete-case set. The label remains indeterminate if an arm needed for the classification is unresolved, even when another arm was strong enough to advance the site.

6.3 Patterns that would weaken the mechanism claim

The reusable-transition interpretation weakens if exact replays fail to reproduce the attribution ranking, if success is limited to the discovery renderer, if unrelated-donor and random controls

approach the active CTF, or if the shuffled donor fails to reproduce it. A patch that copies donor-specific fields indicates leakage rather than clean state transfer. A strong immediate likelihood shift followed by a return to the base answer is reported through the repair index. Length-ODD failure limits any claim to the tested operand lengths.

I interpret the complete pattern across intervention types and data splits, since a single successful patch is insufficient: donor leakage, surface similarity and local repair can produce similar observations.

Failure of all active arms is also compatible with a context-dependent bag of arithmetic heuristics rather than a portable schoolbook state (Nikankin et al., 2025). Under that account, overlapping digit-, magnitude- and carry-sensitive rules can produce a correct suffix while no selected residual site contains an interchangeable tuple s_t . A whole-state patch may then fail because later computation rereads the operands, because the relevant rule is distributed across sites, or because the replacement disrupts several context-specific rules at once. This pattern would motivate stratified component ablations; by itself it would not establish the heuristic account.

7. Limits of reasoning-circuit editing

7.1 Localisation and editing

A site that carries a state need not be the site that computes the transition. Later layers may reconstruct an edited value from the prompt or previous tokens. A strong causal effect at one activation also does not identify a safe location for a persistent weight change; localisation and editing can prefer different layers (Hase et al., 2023).

Whole-vector replacement is correspondingly a test of sufficiency at a broad interface; it makes no minimal-circuit claim. It can preserve useful donor covariance, but it can also transfer nuisance features or create a recipient-donor hybrid that is unusual conditional on the recipient prefix. I would therefore narrow a passing site with component, path or supervised-subspace interventions before any claim about where the state is represented.

Target fidelity and collateral change should be viewed together. For a family of edits Θ , the reported set is

$$\mathcal{P} = \{(\text{CTF}_{\text{target}}(\theta), \text{CKL}(\theta)) : \theta \in \Theta\},$$

where the first coordinate is maximised and the second is minimised. This protocol does not combine them into one score.

7.2 Sources of collateral effects

The coherence calculation in Section 3.4 describes one local source of interference. But other sources remain even for orthogonal directions: a state feature may be used by several tasks; another pathway may rederive the state; different renderers may change which components read it; and a mathematically simple replacement may still be off the model’s activation manifold. Donor leakage and collateral KL measure parts of this problem but cannot certify global safety.

7.3 Scope of the edit

The intervention I implement changes an activation during one generated trace. It is conditional and reversible. It does not alter the transition rule in the weights. A persistent edit would require a broader regression suite, a map of downstream consumers and repair paths, and evidence across unseen states and contexts. Those requirements are outside the present study.

8. Discussion

8.1 What the factorial result can show

The four intervention cells allow several outcomes without presuming that the visible trace is either fully causal or fully post-hoc. A hidden-only effect would show that a residual activation can control the tested transition while the printed state is held fixed. A visible-only effect would show that forcing the token can steer later computation; it would not establish natural reliance on that token or redundancy of hidden state. A joint-only effect would show that the two interventions have to agree in the tested setting. Failure of all three active arms would leave open the possibility that the relevant state is elsewhere, distributed more broadly, rapidly repaired, or not organised as a portable state because arithmetic is implemented by context-specific heuristics.

But these outcomes concern the selected layer, timing and intervention family; they do not by themselves locate a unique representation or decompose all paths from the prompt to the answer. The interaction contrast is useful for detecting non-additivity, but it should not be described as a complete causal channel decomposition.

8.2 Repair and distribution shift

Repair matters here because an edited carry can be recomputed from the original operands. If the intervention initially favours the counterfactual transition and later returns to the base answer, the site was causally effective for a short period but did not have final authority. Reporting only the answer would miss this sequence. Reporting only the immediate logit would overstate it.

The OOD splits serve a different purpose. Surface OOD changes both opening wording and instruction-line layout while preserving the output schema, and checks dependence on that renderer; carry-pattern OOD checks unseen transition combinations; and length OOD checks the tested computation depth. Failure on one split limits the corresponding generalisation claim. IID and OOD results are therefore reported separately.

8.3 Relation to chain-of-thought monitoring

Visible reasoning can provide useful evidence about a model’s computation, but it need not contain all answer-relevant state. It can also change the computation by providing tokens that later positions attend to. Transcript perturbations and internal interventions should therefore be interpreted together. The reviewed evidence is consistent with a mixed account: some intermediate states support reusable transitions, while other parts of a trace provide memory, routing cues or later explanations. I estimate these roles separately for addition.

9. Limitations

The protocol requires an exact high-level state and counterfactual interpreter. Addition provides both; open-ended theorem proving or scientific reasoning may not. Conclusions from this task should not be extended to those settings without a new state definition and validation procedure. The required JSONL trace may itself elicit a serial scratchpad procedure the model would not use for a bare addition prompt, where heuristic mechanisms may instead dominate (Nye et al., 2021; Nikankin et al., 2025); conclusions are therefore scoped to how the model executes this elicited structured-addition procedure, not to natural arithmetic.

An exact replay can still transfer nuisance information or produce an activation that is unusual in the recipient context. Matching, shuffled donors, unrelated donors, random controls and leakage metrics reduce this concern but cannot remove it. The implementation patches a complete

residual vector at a selected block output. A pass establishes operational sufficiency of that donor vector in the hybrid recipient context; it does not identify the named field as the transported cause, isolate attention heads or MLPs, or recover a minimal subspace.

Forced visible tokens alter the autoregressive context. This is intentional for testing visible-token authority, but hidden-only and token-only conditions are not perfectly symmetric physical interventions. Their effects are therefore reported as factorial response contrasts rather than a full mediation analysis.

My review is selective rather than systematic; several cited works are also preprints or laboratory reports. No baseline or intervention run has been completed under this protocol. Inaccessible models, missing activations, invalid outputs and interrupted runs are recorded in the artifact ledger as diagnostic records and are not converted into model evidence.

10. Conclusion

I have specified a causal test for intermediate-state authority in multi-digit addition. The test defines an exact semantic counterfactual, replaces a complete donor residual vector in hidden arms and checks whether the remaining trace follows the corresponding computation. Hidden-only, visible-token-only and joint interventions are compared with matched donor and random controls, with hidden patches placed at different positions around state-token emission.

Several outcomes are possible. A hidden-state edit may be sufficient while the visible trace is held fixed; a token edit may affect later computation through the generated context; the two representations may be jointly required; or neither edit may control the transition. These outcomes describe intervention responses and should not be treated as a complete causal decomposition without further mediation analysis.

Existing work motivates exact replay and held-out validation but does not answer the present question. The study I specify is designed to supply that evidence.

Research and model testing are ongoing. Until a run passes the declared baseline, completion and provenance checks, this remains a protocol paper.

References

- Ameisen, E., Lindsey, J., Pearce, A., Gurnee, W., Turner, N. L., Chen, B., Citro, C., et al. (2025). Circuit tracing: Revealing computational graphs in language models. *Transformer Circuits Thread*. <https://transformer-circuits.pub/2025/attribution-graphs/methods.html>
- Baherwani, V., Goldstein, T., & Panda, A. (2026). Not all LLM reasoning is visible in the chain-of-thought. *arXiv:2607.22925*. <https://arxiv.org/abs/2607.22925>
- Bertolazzi, L., Mondorf, P., Plank, B., & Bernardi, R. (2025). The validation gap: A mechanistic analysis of how language models compute arithmetic but fail to validate it. *Proceedings of EMNLP 2025*. <https://aclanthology.org/2025.emnlp-main.1495/>
- Bogdan, P. C., Macar, U., Nanda, N., & Conmy, A. (2025). Thought anchors: Which LLM reasoning steps matter? *NeurIPS 2025 Mechanistic Interpretability Workshop* (spotlight). <https://arxiv.org/abs/2506.19143v4>
- Boppana, S., Ma, A., Loeffler, M., Sarfati, R., Bigelow, E., Geiger, A., Lewis, O., & Merullo, J. (2026). Reasoning theater: Disentangling model beliefs from chain-of-thought. *Proceedings of the 43rd International Conference on Machine Learning, PMLR 306*. <https://arxiv.org/abs/2603.05488>

- Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N. L., et al. (2023). Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*. <https://transformer-circuits.pub/2023/monosemantic-features>
- Brinkmann, J., Sheshadri, A., Levoso, V., Swoboda, P., & Bartelt, C. (2024). A mechanistic analysis of a transformer trained on a symbolic multi-step reasoning task. *Findings of the Association for Computational Linguistics: ACL 2024*, 4082-4102. <https://arxiv.org/abs/2402.11917>
- Cabannes, V., Arnal, C., Bouaziz, W., Yang, A., Charton, F., & Kempe, J. (2024). Iteration head: A mechanistic study of chain-of-thought. *Advances in Neural Information Processing Systems*. <https://arxiv.org/abs/2406.02128>
- Chan, L., Garriga-Alonso, A., Goldowsky-Dill, N., Greenblatt, R., Nitishinskaya, J., Radhakrishnan, A., Shlegeris, B., et al. (2022). Causal scrubbing: A method for rigorously testing interpretability hypotheses. *Alignment Forum*. <https://www.alignmentforum.org/posts/JvZhhzycHu2Yd57RN/causal-scrubbing-a-method-for-rigorously-testing>
- Chen, X., Plaat, A., & van Stein, N. (2026). How does chain of thought think? Mechanistic interpretability of chain-of-thought reasoning with sparse autoencoding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30297-30305. <https://ojs.aaai.org/index.php/AAAI/article/view/40281>
- Chen, Y., Benton, J., Radhakrishnan, A., Uesato, J., Denison, C., Schulman, J., Somani, A., et al. (2025). Reasoning models don’t always say what they think. *arXiv:2505.05410*. <https://arxiv.org/abs/2505.05410>
- Conmy, A., Mavor-Parker, A. N., Lynch, A., Heimersheim, S., & Garriga-Alonso, A. (2023). Towards automated circuit discovery for mechanistic interpretability. *Advances in Neural Information Processing Systems*, 36. <https://arxiv.org/abs/2304.14997>
- Cunningham, H., Ewart, A., Riggs, L., Huben, R., & Sharkey, L. (2023). Sparse autoencoders find highly interpretable features in language models. *International Conference on Learning Representations*. <https://arxiv.org/abs/2309.08600>
- Dunefsky, J., Chlenski, P., & Nanda, N. (2024). Transcoders find interpretable LLM feature circuits. *Advances in Neural Information Processing Systems*. <https://arxiv.org/abs/2406.11944>
- Dutta, S., Singh, J., Chakrabarti, S., & Chakraborty, T. (2024). How to think step-by-step: A mechanistic understanding of chain-of-thought reasoning. *Transactions on Machine Learning Research*. <https://arxiv.org/abs/2402.18312>
- Elhage, N., Hume, T., Olsson, C., Schiefer, N., Henighan, T., Kravec, S., Hatfield-Dodds, Z., et al. (2022). Toy models of superposition. *Transformer Circuits Thread; arXiv:2209.10652*. <https://arxiv.org/abs/2209.10652>
- Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., et al. (2021). A mathematical framework for transformer circuits. *Transformer Circuits Thread*. <https://transformer-circuits.pub/2021/framework/index.html>
- Feng, J., & Steinhardt, J. (2024). How do language models bind entities in context? *International Conference on Learning Representations*. <https://arxiv.org/abs/2310.17191>
- Gao, L., Dupré la Tour, T., Tillman, H., Goh, G., Troll, R., Radford, A., Sutskever, I., Leike, J., & Wu, J. (2024). Scaling and evaluating sparse autoencoders. *International Conference on Learning Representations*. <https://arxiv.org/abs/2406.04093>

- Geiger, A., Ibeling, D., Zur, A., Chaudhary, M., Chauhan, S., Huang, J., Arora, A., Wu, Z., et al. (2025). Causal abstraction: A theoretical foundation for mechanistic interpretability. *Journal of Machine Learning Research*, 26. <https://arxiv.org/abs/2301.04709>
- Geiger, A., Lu, H., Icard, T., & Potts, C. (2021). Causal abstractions of neural networks. *Advances in Neural Information Processing Systems*, 34. <https://arxiv.org/abs/2106.02997>
- Geiger, A., Wu, Z., Lu, H., Rozner, J., Kreiss, E., Icard, T., Goodman, N. D., & Potts, C. (2022). Inducing causal structure for interpretable neural networks. *Proceedings of the 39th International Conference on Machine Learning, PMLR 162*. <https://arxiv.org/abs/2112.00826>
- Geiger, A., Wu, Z., Potts, C., Icard, T., & Goodman, N. D. (2024). Finding alignments between interpretable causal variables and distributed neural representations. *Proceedings of the Conference on Causal Learning and Reasoning*. <https://arxiv.org/abs/2303.02536>
- Goldowsky-Dill, N., MacLeod, C., Sato, L., & Arora, A. (2023). Localizing model behavior with path patching. *arXiv:2304.05969*. <https://arxiv.org/abs/2304.05969>
- Goyal, S., Ji, Z., Rawat, A. S., Menon, A. K., Kumar, S., & Nagarajan, V. (2024). Think before you speak: Training language models with pause tokens. *International Conference on Learning Representations*. <https://arxiv.org/abs/2310.02226>
- Hanna, M., Liu, O., & Variengien, A. (2023). How does GPT-2 compute greater-than? Interpreting mathematical abilities in a pre-trained language model. *Advances in Neural Information Processing Systems*, 36. <https://arxiv.org/abs/2305.00586>
- Hanna, M., Pezzelle, S., & Belinkov, Y. (2024). Have faith in faithfulness: Going beyond circuit overlap when finding model mechanisms. *First Conference on Language Modeling (COLM 2024)*. <https://arxiv.org/abs/2403.17806>
- Hao, S., Sukhbaatar, S., Su, D., Li, X., Hu, Z., Weston, J., & Tian, Y. (2024). Training large language models to reason in a continuous latent space. *International Conference on Learning Representations*. <https://arxiv.org/abs/2412.06769>
- Hase, P., Bansal, M., Kim, B., & Ghandeharioun, A. (2023). Does localization inform editing? Surprising differences in causality-based localization vs. knowledge editing in language models. *Advances in Neural Information Processing Systems*, 36. <https://arxiv.org/abs/2301.04213>
- Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., & Iwasawa, Y. (2022). Large language models are zero-shot reasoners. *Advances in Neural Information Processing Systems*, 35. <https://arxiv.org/abs/2205.11916>
- Lanham, T., Chen, A., Radhakrishnan, A., Steiner, B., Denison, C., Hernandez, D., Li, D., et al. (2023). Measuring faithfulness in chain-of-thought reasoning. *arXiv:2307.13702*. <https://arxiv.org/abs/2307.13702>
- Leask, P., Bussmann, B., Pearce, M., Bloom, J., Tigges, C., Al Moubayed, N., Sharkey, L., & Nanda, N. (2025). Sparse autoencoders do not find canonical units of analysis. *International Conference on Learning Representations*. <https://arxiv.org/abs/2502.04878>
- Levy, A. A., & Geva, M. (2025). Language models encode numbers using digit representations in base 10. *Proceedings of NAACL 2025: Human Language Technologies, Volume 2 (Short Papers)*, 385-395. <https://aclanthology.org/2025.naacl-short.33/>
- Li, Z., Liu, H., Zhou, D., & Ma, T. (2024). Chain of thought empowers transformers to solve inherently serial problems. *International Conference on Learning Representations*. <https://arxiv.org/abs/2402.12875>

- Liang, J., & Pan, L. (2026). Do latent-CoT models think step-by-step? A mechanistic study on sequential reasoning tasks. *arXiv:2602.00449*. <https://arxiv.org/abs/2602.00449>
- Lieberum, T., Rahtz, M., Kramár, J., Nanda, N., Irving, G., Shah, R., & Mikulik, V. (2023). Does circuit analysis interpretability scale? Evidence from multiple choice capabilities in Chinchilla. *arXiv:2307.09458*. <https://arxiv.org/abs/2307.09458>
- Lindsey, J., Gurnee, W., Ameisen, E., Chen, B., Pearce, A., Turner, N. L., Citro, C., et al. (2025). On the biology of a large language model. *Transformer Circuits Thread*. <https://transformer-circuits.pub/2025/attribution-graphs/biology.html>
- Macar, U., Bogdan, P. C., Rajamanoharan, S., & Nanda, N. (2026). Thought branches: Interpreting LLM reasoning requires resampling. *International Conference on Learning Representations*. <https://arxiv.org/abs/2510.27484>
- Makelov, A., Lange, G., & Nanda, N. (2025). Towards principled evaluations of sparse autoencoders for interpretability and control. *International Conference on Learning Representations*. <https://arxiv.org/abs/2405.08366>
- Marks, S., Rager, C., Michaud, E. J., Belinkov, Y., Bau, D., & Mueller, A. (2025). Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. *International Conference on Learning Representations*. <https://arxiv.org/abs/2403.19647>
- McGrath, T., Rahtz, M., Kramár, J., Mikulik, V., & Legg, S. (2023). The Hydra effect: Emergent self-repair in language model computations. *arXiv:2307.15771*. <https://arxiv.org/abs/2307.15771>
- Mehrafarin, H., Parekh, A., & Konstas, I. (2026). When chain-of-thought fails, the solution hides in the hidden states. *Findings of the Association for Computational Linguistics: EMNLP 2026* (to appear). <https://arxiv.org/abs/2604.23351>
- Meng, K., Bau, D., Andonian, A., & Belinkov, Y. (2022). Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 35. <https://arxiv.org/abs/2202.05262>
- Miller, J., Chughtai, B., & Saunders, W. (2024). Transformer circuit faithfulness metrics are not robust. *Conference on Language Modeling*. <https://arxiv.org/abs/2407.08734>
- Mueller, A., Geiger, A., Wiegrefe, S., Arad, D., Arcuscin, I., Belfki, A., Chan, Y. S., et al. (2025). MIB: A mechanistic interpretability benchmark. *Proceedings of the 42nd International Conference on Machine Learning, PMLR 267*. <https://arxiv.org/abs/2504.13151>
- Nanda, N. (2023). Attribution patching: Activation patching at industrial scale. *LessWrong / Alignment Forum*. <https://www.lesswrong.com/posts/gtLLBhzQTG6nKTeCZ/attribution-patching-activation-patching-at-industrial-scale>
- Nanda, N., Chan, L., Lieberum, T., Smith, J., & Steinhardt, J. (2023). Progress measures for grokking via mechanistic interpretability. *International Conference on Learning Representations*. <https://arxiv.org/abs/2301.05217>
- Nikankin, Y., Reusch, A., Mueller, A., & Belinkov, Y. (2025). Arithmetic without algorithms: Language models solve math with a bag of heuristics. *International Conference on Learning Representations*. <https://arxiv.org/abs/2410.21272>
- Nosek, B. A., Ebersole, C. R., DeHaven, A. C., & Mellor, D. T. (2018). The preregistration revolution. *Proceedings of the National Academy of Sciences*, 115(11), 2600-2606. <https://doi.org/10.1073/pnas.1708274114>
- Nye, M., Andreassen, A. J., Gur-Ari, G., Michalewski, H., Austin, J., Bieber, D., Dohan, D., et al. (2021). Show your work: Scratchpads for intermediate computation with language models.

- ICLR 2022 Workshop on Deep Learning for Code (DL4C)*. <https://arxiv.org/abs/2112.00114>
- Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., et al. (2022). In-context learning and induction heads. *Transformer Circuits Thread*. <https://arxiv.org/abs/2209.11895>
- Park, K., Choe, Y. J., & Veitch, V. (2024). The linear representation hypothesis and the geometry of large language models. *Proceedings of the 41st International Conference on Machine Learning, PMLR 235*, 39643-39666. <https://proceedings.mlr.press/v235/park24c.html>
- Pfau, J., Merrill, W., & Bowman, S. R. (2024). Let’s think dot by dot: Hidden computation in transformer language models. *First Conference on Language Modeling (COLM 2024)*. <https://arxiv.org/abs/2404.15758>
- Quirke, P., & Barez, F. (2024). Understanding addition in transformers. *International Conference on Learning Representations*. <https://arxiv.org/abs/2310.13121>
- Rajamanoharan, S., Lieberum, T., Sonnerat, N., Conmy, A., Varma, V., Kramár, J., & Nanda, N. (2024). Jumping ahead: Improving reconstruction fidelity with JumpReLU sparse autoencoders. *arXiv:2407.14435*. <https://arxiv.org/abs/2407.14435>
- Shen, Z., Yan, H., Zhang, L., Hu, Z., Du, Y., & He, Y. (2025). CODI: Compressing chain-of-thought into continuous space via self-distillation. *Proceedings of EMNLP 2025*, 677-693. <https://arxiv.org/abs/2502.21074>
- Shih, B., Winnicki, J., & Darve, E. (2026). When does activation steering change what a model computes from? *arXiv:2606.29522*. <https://arxiv.org/abs/2606.29522>
- Stolfo, A., Belinkov, Y., & Sachan, M. (2023). A mechanistic interpretation of arithmetic reasoning in language models using causal mediation analysis. *Proceedings of EMNLP 2023*. <https://arxiv.org/abs/2305.15054>
- Syed, A., Rager, C., & Conmy, A. (2024). Attribution patching outperforms automated circuit discovery. *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, 407-416. <https://arxiv.org/abs/2310.10348>
- Templeton, A., Conerly, T., Marcus, J., Lindsey, J., Bricken, T., Chen, B., Pearce, A., et al. (2024). Scaling monosemanticity: Extracting interpretable features from Claude 3 Sonnet. *Transformer Circuits Thread*. <https://transformer-circuits.pub/2024/scaling-monosemanticity/>
- Tishby, N., Pereira, F. C., & Bialek, W. (1999). The information bottleneck method. *37th Annual Allerton Conference on Communication, Control, and Computing*. <https://arxiv.org/abs/physics/0004057>
- Turpin, M., Michael, J., Perez, E., & Bowman, S. R. (2023). Language models don’t always say what they think: Unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems*, 36. <https://arxiv.org/abs/2305.04388>
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30. <https://arxiv.org/abs/1706.03762>
- Vig, J., Gehrmann, S., Belinkov, Y., Qian, S., Nevo, D., Singer, Y., & Shieber, S. (2020). Causal mediation analysis for interpreting neural NLP: The case of gender bias. *arXiv:2004.12265*. <https://arxiv.org/abs/2004.12265>
- Wang, K. R., Variengien, A., Conmy, A., Shlegeris, B., & Steinhardt, J. (2023). Interpretability in the wild: A circuit for indirect object identification in GPT-2 small. *International Conference on Learning Representations*. <https://arxiv.org/abs/2211.00593>

- Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q. V., & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35. <https://arxiv.org/abs/2201.11903>
- Welch, L. R. (1974). Lower bounds on the maximum cross correlation of signals. *IEEE Transactions on Information Theory*, 20(3), 397-399. <https://doi.org/10.1109/TIT.1974.1055219>
- Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., et al. (2025). Qwen2.5 technical report. *arXiv:2412.15115v2*. <https://arxiv.org/abs/2412.15115v2>
- Yuan, A., Su, Z. J., Zhang, H., Nian, Y., & Zhao, Y. (2026). Hidden error awareness in chain-of-thought reasoning: The signal is diagnostic, not causal. *Mechanistic Interpretability Workshop at ICML 2026*. <https://arxiv.org/abs/2605.09502>
- Zhang, L., Dong, W., Zhang, Z., Yang, S., Hu, L., Liu, N., Zhou, P., & Wang, D. (2025). EAP-GP: Mitigating saturation effect in gradient-based automated circuit identification. *Advances in Neural Information Processing Systems*, 38. <https://arxiv.org/abs/2502.06852>
- Zhong, Z., Liu, Z., Tegmark, M., & Andreas, J. (2023). The clock and the pizza: Two stories in mechanistic explanation of neural networks. *Advances in Neural Information Processing Systems*, 36. <https://arxiv.org/abs/2306.17844>
- Zhu, F., Wang, P., & Sui, Z. (2025). Chain-of-thought tokens are computer program variables. *arXiv:2505.04955*. <https://arxiv.org/abs/2505.04955>