PROGRAMMABLE SEMANTIC INTERVENTIONS: LEARNING CAUSAL CARRIERS FROM BEHAVIORAL PROGRAMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Causal abstraction explains a frozen network by aligning its activations with the variables of a simpler, interpretable model, and checking that interventions on the two agree. Finding such an alignment currently requires a counterfactual dataset: for every pair of prompts, the answer the network ought to give once the variable has been changed. Producing those labels means committing to an executable algorithm for the behavior before studying the model supposed to implement it. However, the criterion these methods optimize only leverages the answer an intervention should produce. Here, we develop an approach that lets the investigator define that relationship directly: which answers the variable is about, and how they should move when it changes. We call this interface a *semantic program* and introduce *Programmable Semantic Interventions* (PSI, Ψ): a low-rank activation subspace learned so that interchanging it moves the model's answer the way the program declares. On the MIB benchmark Ψ matches or surpasses the quality of supervised alignment, while requiring minimal supervision. The declaration also fixes a coordinate system with known semantics, enabling donor-free steering: jointly training Ψ for both localization and control realizes 94.8% of held-out requests for answers the investigator chooses, without the need for donor prompts.

1 INTRODUCTION

Mechanistic interpretability sets out to explain the inner workings of a network in terms of the algorithm it implements (Olah et al., 2020; Sharkey et al., 2025). Recently, an increased emphasis has been placed on providing *causal* claims (Mueller et al., 2024), i.e., what part of a network's activations encodes an abstract, high-level (simpler) variable, and does an intervention move the network's behavior in a predictable way that the corresponding high-level model says it should (Geiger et al., 2025)? The latter distinction is essential: reading a property off an activation with a trained probe establishes that the property is present (Belinkov, 2022), while a causal claim asks whether the network actually uses it. Namely, causal abstraction makes the causal claim testable (Geiger et al., 2021; 2025), and standard benchmarks now exist to evaluate methods against it (Arora et al., 2024; Mueller et al., 2025). Intervening on activations to attribute behavior is now standard practice (Vig et al., 2020; Meng et al., 2022), and hand-built circuit analyses show what a complete high-level account of a single behavior costs to produce (Wang et al., 2023). However, testing *causal claims* at scale currently requires the construction of a counterfactual dataset. For every pair of inputs differing (only) in the value of a putative high-level variable, one must build the answer the network ought to give once that value has been exchanged. Methods in the distributed alignment lineage obtain that answer by executing a high-level algorithm the investigator has specified in advance (Geiger et al., 2024; Wu et al., 2023; Huang et al., 2024; Sun et al., 2025). The investigator, therefore, commits to an algorithm for the behavior before studying the model that is supposed to implement it, and the approach reaches only those behaviors for which such an algorithm can be written down, using the prompt pairs curated by someone.

The success criterion for identification is *interchange intervention accuracy* (Geiger et al., 2022), and current methods are directly optimized against it. However, the criterion itself never inspects the high-level algorithm: it merely relies on the answer the algorithm produces under the intervention and compares it with the intervened network's output. This simple observation points to

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

a significant gap between the amount of required specifications and what is ultimately used. Concretely, consider the problem of localizing the variable encoding the position of the correct option in a four-choice question. A full high-level causal specification, required by current methods, is an executable algorithm that employs tens of (unobserved) variables, each with their domain, interaction edges between them and mechanisms to resolve their value, including the one that ultimately scans the options to formulate the answer. We propose a different interface – semantic programs – that only asks the investigator to declare what the criterion consumes.

A *semantic program* names the answers a variable is about and states how they should move when the variable is set; everything else is measured from the frozen model. We call the method built on it *Programmable Semantic Interventions* (PSI, Ψ): a low-rank activation subspace, learned so that interchanging it moves the answer the way the program says. Moreover, a program written in the model's own answer space induces a coordinate system whose semantics the investigator knows, such that an arbitrary behavior can be directly *requested* in it. Coming back to the question-answering example, a semantic program for the same task consists of just two lines: a four-option library of labels, plus the rule that an interchange must replace the base's label with the source's. Fundamentally, a full high-level specification is an artifact capable of sitting the exam and producing the correct answer, while a semantic program only has to name the possible outcomes.

Related work. Work on localizing causal variables has steadily reduced the choices that an investigator makes by hand. Distributed alignment search learns the subspace by gradient descent (Geiger et al., 2024), its boundless variant learns the rank along with it (Wu et al., 2023), multi-task training adds attribute isolation (Huang et al., 2024), and a hypernetwork takes over the search across token positions (Sun et al., 2025). However, throughout this progression, the requirement of an executable high-level model remains, and each method trains on a curated list of prompt pairs. Ψ thus follows this lineage by proposing a novel specification rather than adding more automation machinery. A second line of work encodes a behavior into activations with no donor example. Directions fitted from contrastive prompts steer a model at inference (Zou et al., 2023; Turner et al., 2023; Rimsky et al., 2024; Li et al., 2023), and recent methods learn the write operator (Bao et al., 2026; Rodriguez et al., 2025) or amortize it over a space of requests (Shi et al., 2026). What such a method writes is a direction with a fitted magnitude, selected to elicit a behavior. A *semantic program* writes instead a distribution over a declared answer set indexed by a causal variable's values, into a subspace that interchange interventions have validated first. Localizing a variable and controlling it are known to be distinct capabilities: the directions that read a concept and the directions that change it are geometrically distinct (Park et al., 2024; Galeone et al., 2026; Wu et al., 2025), steering vectors are brittle across prompts (Tan et al., 2024), and a subspace can move an output through a pathway unrelated to the feature it names (Makelov et al., 2024). We thus validate both capabilities in Section 3 and refer to Appendix A.1 for a broader overview of the relevant literature.

Contributions. We introduce the semantic program as the interface that a causal-variable method requests from the investigator, and Ψ, which learns a carrier from it. Three results follow: (*i*) on the causal-variable localization tasks, Ψ reaches the baseline's quality without a counterfactual dataset, and is ahead of it in most of the cells we have collected; (*ii*) ablating the supervision, we measure the value of a counterfactual dataset and find that a method requiring no pair list recovers most of the performance by benefiting from a larger donor pool size which provides observations the pair list cannot reach; (*iii*) finally, because the declaration is also an address space, the same carrier realizes readings that the investigator requests, with no source prompt and including readings that no prompt in the data exhibits, surpassing what an interchange from a real donor achieves at the same site.

2 METHODS

Problem setup. We consider a frozen subject ϕ, an input prompt $\mathbf{x}$, and indicate with $\phi^n(\mathbf{x})$ the standard auto-regressive greedy rollout of the model predictions for a sentence length n. We indicate by $\boldsymbol{h} = \phi|_\kappa(\mathbf{x})$ the hidden vector of activations $\boldsymbol{h} \in \mathbb{R}^d$ produced by the model, where κ is a set of location indices (e.g. the collection of token positions and layers in the residual stream of a transformer network). *Causal abstraction* (Geiger et al., 2025) formalizes what it means for such a vector to carry a concept: an abstraction $\mathcal{A}$ maps the low-level states of the network to the states of a high-level model $\mathbf{H}$, whose variables are the concepts we wish to reason with, and it is faithful under a commutative condition on interventions—intervening on the low-level model and abstracting the

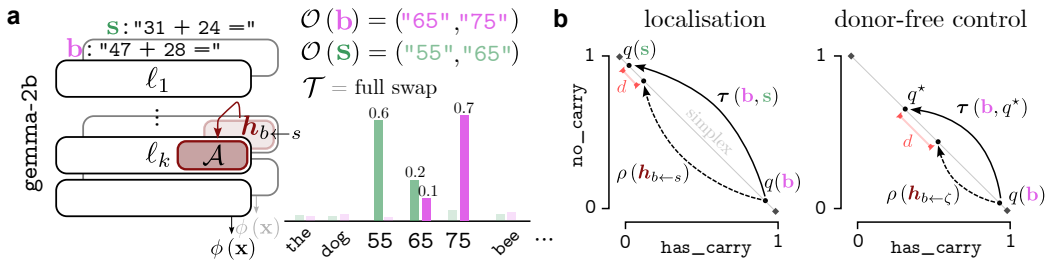

Figure 1: **Programmable Semantic Interventions. (a)** The investigator declares which answers $\mathcal{O}$ the variable is about: here the two numerical sums a carry does and does not produce, which result in different strings for the base $\mathbf{b}$ and the source $\mathbf{s}$. The frozen subject supplies its probability on each of them, read at the cut ℓ_k where the abstraction $\mathcal{A}$ selects the carrier. **(b)** A reading is a point in the simplex over that library. Interchanging the source's carrier should move the base's reading to $\boldsymbol{\tau}(\mathbf{b},\mathbf{s})$, and the distance d to what the intervened model reads is the training signal; the same target can be named directly as $q^\star$ and realized by an edit, with no source prompt.

result should agree with abstracting first and then performing the *corresponding* intervention on $\mathbf{H}$ (see Geiger et al., 2025 for the formal exposition). In our experiments, $\mathcal{A} : \mathbb{R}^d \to \mathbb{R}^r$ is a linear map from the hidden vector $\boldsymbol{h}$ to an r-dimensional high-level representation $\boldsymbol{z} = \mathcal{A}(\boldsymbol{h})$, and an *intervention* is performed by replacing part of the hidden vector $\boldsymbol{h}_b = \phi|_\kappa(\mathbf{b})$ of a base prompt $\mathbf{b}$ with the corresponding part of the hidden vector $\boldsymbol{h}_s = \phi|_\kappa(\mathbf{s})$ of a source prompt $\mathbf{s}$, and then measuring the resulting output $\phi^n_{b\leftarrow s}(\mathbf{b})$.

Measuring abstraction quality. Empirically, one quantifies the faithfulness of a candidate abstraction $\mathcal{A}$ by measuring the agreement between the two sides of the commutative condition over a distribution of interventions. An intervention is a pair $\iota = (\mathbf{b},\mathbf{s})$ of a base and a source prompt, drawn from $\mathcal{B}\times\mathcal{S}$. The *interchange intervention accuracy* (IIA) (Geiger et al., 2022; 2024) has been proposed as one such criterion: it measures how often the intervened low-level model and the high-level model produce the same output. Both sides of that comparison are answers in the subject's own vocabulary, allowing for direct comparison, and we can write

$$\text{IIA} \;=\; \mathbb{E}_{\iota\sim\mathcal{B}\times\mathcal{S}}\Big[\,\mathbf{1}\big[\,\phi^n_{b\leftarrow s}(\mathbf{b}) \;=\; \mathbf{H}_{b\leftarrow s}(\mathbf{b})\,\big]\,\Big], \tag{1}$$

where, with a slight abuse of notation, $\mathbf{H}_{b\leftarrow s}(\mathbf{b})$ denotes the high-level model's output under the intervention corresponding to $b \leftarrow s$, and the indicator $\mathbf{1}[\cdot]$ evaluates to 1 when the two outputs agree and 0 otherwise.

Aligning a subspace. Replacing the hidden vector wholesale would test whether the *entire* activation carries the concept (the *Full Vector* baseline of the MIB suite, Mueller et al., 2025), which is a rather crude baseline: a single residual stream is expected to carry many concurrent variables, more of them than it has dimensions (Elhage et al., 2022). A common alternative is to let the abstraction single out a low-dimensional subspace (Geiger et al., 2024) or a sparse set of learned features (Cunningham et al., 2024; Gao et al., 2025), leaving the rest of the activation untouched. Concretely, in our work we take an orthonormal $A \in \mathbb{R}^{r\times d}$ with $r \ll d$, write $\boldsymbol{\pi} = A^\top A$ for the projector onto its row space, and define the intervention that transplants the source's *carrier* into the base while preserving the base's complement:

$$\boldsymbol{h}_{b\leftarrow s} = \boldsymbol{\pi}\boldsymbol{h}_s + (\mathrm{I} - \boldsymbol{\pi})\boldsymbol{h}_b. \tag{2}$$

The abstraction is then simply the readout of that subspace, $\mathcal{A}(\boldsymbol{h}) = A\boldsymbol{h}$, and identifying the causal variable amounts to *finding* A. Nothing in the framework requires $\mathcal{A}$ to be linear, and a richer abstraction would fit a wider class of high-level models. However, in our experiments we restrict ourselves to linear maps as non-linear alignments weaken what a successful abstraction certifies (Sutter et al., 2025). Moreover, a linear carrier keeps the intervention of Eq. (2) a projection, so the edit has a scale we can compare against the displacements interchange interventions naturally produce.

What the IIA criterion requires. One observation about Eq. (1) motivates our interface proposal (Section 2.1). The IIA criterion never inspects the inner workings of $\mathbf{H}$. It reads a single quantity off the high-level model: the output it produces under the intervention, while remaining oblivious to its variables, its mechanisms, or its causal graph. Existing methods (Geiger et al., 2024; Wu et al., 2023; Huang et al., 2024; Sun et al., 2025) nonetheless specify $\mathbf{H}$ in full, as an executable algorithm, and obtain that output by running it. The result of that computation is then taken as the counterfactual label attached to each training pair. Thus, the full specification of $\mathbf{H}$ is typically taken as a prerequisite for causal investigation. In particular, it requires postulating a particular algorithm the low-level model ϕ is supposed to implement before one has studied the model itself, and it confines the method to tasks for which a full algorithmic specification is available.

2.1 Programmable Semantic Interventions

Our proposal is to pay only for what the criterion reads. Rather than specifying the high-level model $\mathbf{H}$ fully and computing its output, the investigator *declares* that output directly, as a rule stated in the subject's own answer space. This declaration requires two components: an outcome library $\mathcal{O}$ that specifies which answers the concept pertains to, and a transport rule $\mathcal{T}$ that indicates how those answers should be transferred under an intervention. We call the resulting pair of objects that make up this declaration a *semantic program*, $\mathcal{P} = (\mathcal{O}, \mathcal{T})$. We offer a graphical overview of the method in Figure 1.

The outcome library. The outcome library is a small, prompt-dependent set of candidate answers, one for each value the variable of interest can take:

$$\mathcal{O}(\mathbf{x}) \;=\; \big(o_1(\mathbf{x}), \ldots, o_k(\mathbf{x})\big). \tag{3}$$

To better illustrate the role of the library $\mathcal{O}$, we consider the explicit example of two-digit addition, with the carry as the variable of interest (the Arithmetic task of Mueller et al., 2025). For the prompt $\mathbf{x}$ = `"47 + 28 ="`, the units digits sum to `15`, so a carry of one propagates and the correct answer is `75`; had the carry been zero, the answer would have been `65`. The library for this prompt is therefore $\mathcal{O}(\mathbf{x}) = (\texttt{"65"}, \texttt{"75"})$, indexed by the two values of the carry. Similarly, for a different expression $\mathcal{O}$ would correspond to the set of two possible outcomes, with and without a carry. For a multiple-choice question, the library is the four answer symbols, e.g. `A`, `B`, `C`, `D`, indexed by which position holds the correct option; for an entity-attribute lookup (e.g. the RAVEL task of Huang et al., 2024) it is the set of admissible attribute values. We highlight two properties of Eq. (3). First, its entries are answers in the subject's vocabulary: they are the (potentially multi-token) strings the model would have to emit. Second, the library says nothing about which of the library entries is *correct*. Its role is to fix a (semantic) reference frame, i.e. the set of outputs this concept is about and expected to elicit, and to leave the readings within that frame to the model.

Once a frame is declared, the subject ϕ supplies its own coordinates within it. Writing $\rho_i(\mathbf{x}) = \log p_\phi\big(o_i(\mathbf{x}) \mid \mathbf{x}\big)$ for the log-probability the model assigns to the i-th library entry, we define the following two quantities:

$$Z(\mathbf{x}) = \sum_{i=1}^{k} e^{\rho_i(\mathbf{x})}, \qquad q_i(\mathbf{x}) = \frac{e^{\rho_i(\mathbf{x})}}{Z(\mathbf{x})}. \tag{4}$$

The *declared mass* $Z(\mathbf{x}) \in [0, 1]$ answers how much of the model's probability the frame captures at all: typically in our experiments the library is composed of a handful of strings, out of a vocabulary of over a hundred thousand tokens; moreover in principle nothing guarantees the model would assign significant probability to any of the library entries in its natural behavior. $Z(\mathbf{x})$ thus quantifies the coverage of the declared library. The *reading* $q(\mathbf{x}) = [q_1(\mathbf{x}), \ldots, q_k(\mathbf{x})]$ is how the captured mass divides between the declared alternatives, and it is the vector we treat as the model's opinion about the variable. Continuing the carry-addition example, a subject that places $p = 0.70$ on `"75"` and $p = 0.05$ on `"65"` has $Z(\mathbf{x}) = 0.75$ and $q(\mathbf{x}) = (0.07, 0.93)$: three quarters of its mass lies inside the frame, and within the frame it is confident the carry fired. Both numbers are thus measured from the frozen model and require no further investigator input.

The transport rule. The second half of the program states how a reading $q(\mathbf{x})$ should change when the variable is intervened upon. We represent this transformation as a map:

$$\boldsymbol{\tau} = \mathcal{T}\left(q(\mathbf{s}), q(\mathbf{b}), \gamma\right), \tag{5}$$

which takes the source's reading $q(\mathbf{s})$, the base's reading $q(\mathbf{b})$, and an optional per-example gate γ, and returns the reading the base *should* exhibit once the source's value of the variable has been written into it. The source reading $q(\mathbf{s})$ can be measured by a declared query for the variable or a donor prompt (see Appendix A.3).

The simplest and most common rule is the full swap, $\boldsymbol{\tau} = q(\mathbf{s})$. Return to the addition example, and let the base be $\mathbf{b} =$ `"47 + 28 ="` (carry one) and the source $\mathbf{s} =$ `"31 + 24 ="` (carry zero). If A really is the subspace that carries the carry, then transplanting the source's carrier into the base should make the base behave as though no carry had occurred: it should answer `65` rather than `75`. In the base's own reference frame, that is precisely the statement that the base's reading should become the source's reading. Note what makes this well posed: the two prompts have different libraries, $\mathcal{O}(\mathbf{b}) =$ (`"65"`, `"75"`) against $\mathcal{O}(\mathbf{s}) =$ (`"55"`, `"65"`), but both are indexed by the same variable, so a reading transports even though a token does not.

However, not every variable is decisive in every context. Consider an entity-attribute task, where the prompt selects which attribute is being asked about the introduced entity. In such a task, interventions should alter the model's behavior only when the base prompt queries the sought attribute (e.g. in the RAVEL task, intervening on the entity's continent should change the answer to a continent question while leaving a language question untouched). Such cases can be easily covered by a gated rule, $\boldsymbol{\tau} = \gamma(\mathbf{b})q(\mathbf{s}) + (1-\gamma(\mathbf{b}))q(\mathbf{b})$, where $\gamma(\mathbf{b}) \in \{0, 1\}$ records whether the intervened variable is the operative one for that prompt; the target is the source's reading when it is, and the base's own reading, i.e. a no-op, when it is not. More generally, $\mathcal{T}$ may be given as a table specifying, for each pair of source and base values, the reading that should result. We refer to Appendix A.3 for a complete description of the programs used in our experiments.

Semantic programs are weak supervision. A semantic program $\mathcal{P}$ is written before any training and consists of Eqs. (3) and (5): a family of candidate answers and a rule for moving between readings of them. It contains no correct answer, no counterfactually annotated example pair, and no executable high-level model. Everything numerical in the target of Eq. (5) is measured from the frozen subject through Eq. (4). In this sense, the investigator supervision provided by specifying $\mathcal{P}$ is weaker than that of alignment methods that require a labeled counterfactual per pair (Geiger et al., 2024; Wu et al., 2023; Huang et al., 2024; Sun et al., 2025). Appendix Table 2 accounts for what each approach asks of the investigator, and Section 3 measures what each input is worth.

2.2 Learning the alignment from the program

A semantic program $\mathcal{P}$ determines a training signal without any further input. Given a collection of base prompts $\mathcal{B}$, we can construct candidate base-source interventions $\iota = (\mathbf{b}, \mathbf{s})$ by pairing via a random derangement within a group of prompts the program declares comparable. The transport rule of Eq. (5) then produces a target reading $\boldsymbol{\tau}(\mathbf{b}, \mathbf{s})$, while the interchange of Eq. (2) produces an activation $\boldsymbol{h}_{b \leftarrow s}$ whose readings $\rho(\boldsymbol{h}_{b \leftarrow s})$ we can measure. Learning the subspace A is then a matter of making the second agree with the first:

$$\mathcal{L}_{\text{donor}}(A) \;=\; \mathbb{E}_{\mathbf{b},\mathbf{s}}\, d\left(\rho(\boldsymbol{h}_{b \leftarrow s}),\; \boldsymbol{\tau}(\mathbf{b}, \mathbf{s})\right). \tag{6}$$

Equation (6) is a differentiable relaxation of Eq. (1): the indicator that asks whether two outputs match exactly is replaced by a divergence d that measures how far apart two readings are, and the expectation is taken over sampled prompt pairs. One natural choice for d is the cross-entropy of the target against the model's log-probabilities (Geiger et al., 2024). Because the ρ_i are log-probabilities over the whole vocabulary rather than over the library, that quantity decomposes as

$$-\sum_i \tau_i\, \rho_i \;=\; \underbrace{H(\boldsymbol{\tau}) \;+\; \mathrm{KL}\big(\boldsymbol{\tau} \,\big\|\, q(\boldsymbol{h}_{b \leftarrow s})\big)}_{\text{the reading}} \;\; \underbrace{-\,\log Z(\boldsymbol{h}_{b \leftarrow s})}_{\text{the mass}}, \tag{7}$$

where H is the entropy of the target and KL the Kullback–Leibler divergence. The first group asks the intervened model to produce the requested reading; the second asks it to place as much probability as possible *inside the library*. Whether that second incentive is wanted depends on what is being asked of the model, so we leave it as a choice. The CONDITIONAL form drops the mass term

and scores the reading alone; the REFERENCE form targets the mass $Z(\mathbf{b})$ the unintervened prompt already shows, so an intervention is asked to preserve it; the FULL form keeps Eq. (7) as written. We used the FULL form for the Arithmetic localization experiments and CONDITIONAL for the rest (Table 1), while Appendix A.5 measures the impact each choice has on control costs.

2.3 ENABLING CONTROL VIA THE SAME INTERFACE

A trained subspace that minimizes Eq. (6) supports interchange interventions, which realize the reading of whichever source prompt is supplied. But one crucial advantage of the reference frame of Eq. (3) is that it represents a coordinate system with *known* semantics (since the investigator designed it in the first place). Such a coordinate system can thus be used to naturally formulate any representable reading target $q^\star$, while the transport rule $\mathcal{T}$ can then be used to *request* such a reading from the model when exposed to a base prompt $\mathbf{b}$, i.e. construct $\boldsymbol{\tau}^\star = \mathcal{T}\,(q^\star, q(\mathbf{b}), \gamma)$. In particular, we can attach a controller:

$$\boldsymbol{\zeta} \;=\; \mathrm{control}\big(\boldsymbol{\tau}^\star,\; q(\mathbf{b}),\; A\boldsymbol{h}_b\big), \tag{8}$$

which maps a requested reading $\boldsymbol{\tau}^\star$, the subject's current reading $q(\mathbf{b})$, and the base's carrier coordinates $A\boldsymbol{h}_b$ to an edit $\boldsymbol{\zeta} \in \mathbb{R}^r$ expressed in those same carrier coordinates. The steered activation is then $\boldsymbol{h}_{b\leftarrow\zeta} = \boldsymbol{h}_b + A^\top\boldsymbol{\zeta}$: by construction the edit lives entirely inside the identified subspace and leaves the complement $(\mathrm{I} - \boldsymbol{\pi})\boldsymbol{h}_b$ untouched, exactly as the interchange of Eq. (2) does.

The simplest instance, and the one we use in our experiments (although more elaborate manifold-aware write operators are possible, e.g. Wurgaft et al., 2026), is a linear gain driven by the residual between the requested and the current reading. Writing $\mathrm{lo}(\boldsymbol{p}) \in \mathbb{R}^{k-1}$ for the log-odds of a reading against its last entry, $\mathrm{lo}_i(\boldsymbol{p}) = \log p_i - \log p_k$, the gain controller is:

$$\boldsymbol{\zeta} \;=\; G\,\big[\,\mathrm{lo}(\boldsymbol{\tau}^\star) \;-\; \mathrm{lo}\big(q(\mathbf{b})\big)\,\big], \qquad G \in \mathbb{R}^{r\times(k-1)}, \tag{9}$$

so that G is the only learned object and the edit vanishes whenever the request coincides with what the model is already doing. We clip the edit to a fixed budget, $\boldsymbol{\zeta} \leftarrow \boldsymbol{\zeta}\,\min(1,\, c/\|\boldsymbol{\zeta}\|)$, with c set to a high quantile of the displacements that interchange interventions themselves produce at that site (Appendix A.10).

The controller is trained jointly with the subspace, against requests drawn half from the readings other prompts exhibit and half uniformly from the probability simplex, so that it is exercised on targets no prompt in the dataset elicits. With d the same divergence as before, the complete objective is

$$\mathcal{L}(A, G) \;=\; \underbrace{\mathbb{E}_{\mathbf{b},\mathbf{s}}\; d\big(\rho(\boldsymbol{h}_{b\leftarrow s}),\; \boldsymbol{\tau}(\mathbf{b},\mathbf{s})\big)}_{\text{donor: the subspace carries the variable}} \;+\; \lambda\, \underbrace{\mathbb{E}_{\mathbf{b},\,\boldsymbol{\tau}^\star}\; d\big(\rho(\boldsymbol{h}_{b\leftarrow\zeta}),\; \boldsymbol{\tau}^\star(\mathbf{b}, q^\star)\big)}_{\text{control: the subspace is writable}}. \tag{10}$$

The two terms share the abstraction A, which is what makes the training joint: the donor term asks the subspace to behave like the variable under transplantation, and the control term asks the same subspace to be reachable by a learned edit. At inference, the controller of Eq. (8) accepts any distribution over the declared library, including readings $q^\star$ no prompt in the dataset elicited, such as an even split between two answers, with no source prompt and no retraining. This is a capability methods trained on the interchange of Eq. (2) (Geiger et al., 2024; Wu et al., 2023; Huang et al., 2024; Sun et al., 2025) do not offer directly, since their interchange operator requires a donor. In Section 3.2 we therefore equip their frozen carriers with a controller of our own and measure what that recovers.

3 RESULTS

Protocol. We evaluate on the causal-variable tasks of Mueller et al. (2025), with the benchmark's subjects gemma-2_2b, llama-3.1_8b and qwen-2.5_0.5b, and with gpt2-small for IOI. At each site we set the carrier rank to the subspace dimension the official DAS baseline uses there, so the

Table 1: **Localizing causal variables on MIB.** Interchange intervention accuracy (%, higher is better): the mean over the depth profile, with the best layer in parentheses. Ψ uses the rank the DAS baseline uses at each cell. **Bold** is the best method in a column, mean and best layer separately, and the gray rows are nulls that do not compete for it. Baselines from Mueller et al. (2025).

	ARC (Easy)			
	gemma-2_2b		llama-3.1_8b	
Method	O_{Answer}	X_{Order}	O_{Answer}	X_{Order}
Full Vector	63 (100)	43 (74)	68 (100)	47 (72)
Random	18 (27)	33 (33)	17 (28)	33 (33)
DBM	82 (**99**)	63 (80)	85 (**100**)	69 (82)
+ PCA	78 (98)	64 (81)	84 (**100**)	72 (83)
+ SAE	70 (89)	54 (70)	74 (94)	55 (67)
DAS	88 (94)	**76** (88)	88 (99)	74 (84)
Ψ (ours)	**95** (98)	71 (**89**)	**97** (**100**)	**80** (**92**)

	Arithmetic (+)[†]	
	gemma-2_2b	llama-3.1_8b
Method	X_{Carry}	X_{Carry}
Full Vector	29 (35)	35 (45)
Random	26 (26)	26 (26)
DBM	33 (43)	47 (58)
+ PCA	32 (44)	37 (56)
+ SAE	32 (44)	38 (55)
DAS	31 (35)	**54** (**65**)
Ψ (ours)	**38** (**46**)	52 (64)

	MCQA					
	gemma-2_2b		llama-3.1_8b		qwen-2.5_0.5b	
Method	O_{Answer}	X_{Order}	O_{Answer}	X_{Order}	O_{Answer}	X_{Order}
Full Vector	61 (100)	44 (77)	77 (100)	46 (68)	35 (99)	49 (99)
Random	19 (28)	33 (33)	16 (31)	33 (33)	7 (18)	33 (33)
DBM	84 (**99**)	63 (84)	86 (**100**)	66 (73)	46 (94)	60 (99)
+ PCA	57 (96)	52 (81)	65 (99)	53 (74)	22 (76)	54 (**100**)
+ SAE	73 (90)	51 (65)	80 (99)	58 (65)	—	—
DAS	**95** (97)	**77** (**93**)	94 (**100**)	77 (**91**)	86 (95)	**78** (**100**)
Ψ (ours)	94 (**99**)	76 (92)	**98** (**100**)	**80** (**91**)	**92** (**98**)	75 (**100**)

two are compared at equal capacity, and we follow the benchmark's site-selection protocol: at each layer we take the best of the benchmark's fixed token positions, then report the mean of that depth profile together with its best layer. We also include a *random* subspace of the same rank at the same sites as a floor. Localization runs use $\lambda = 0$ in Eq. (10), while the control experiments of Section 3.2 use $\lambda = 1$ and focus on gemma-2_2b at two sites. Reported localization numbers are single-seed, following Mueller et al. (2025) (and due to the expensive evaluation, see Appendix A.9); at the one site (MCQA X_{Order}, gemma-2_2b layer 17, last_token) where we repeated the experiment three times, the spread was ± 0.9 points, comparable to one evaluation pair (0.8 pt).

3.1 Localizing causal variables

Table 1 reports interchange intervention accuracy against the benchmark's published baselines. DAS is the closest method to our approach: it also fits a rotation of the residual stream at a fixed rank, but it is given an executable high-level model and a curated list of base–source pairs whose counterfactual answer is known in advance. Ψ is given neither, and at its best layer it matches or exceeds DAS at all but two cells, trailing by about a point in both. The margin is clearest on the other two subjects: on llama-3.1_8b Ψ gains about four points of depth mean on both MCQA variables (97.8 against 94, and 80.4 against 77) and six on ARC X_{Order} (79.9 against 74), and on qwen-2.5_0.5b it gains six on O_{Answer} (92.1 against 86). ARC O_{Answer} gives the widest margins of all, at 94.8 against 88 on gemma-2_2b and 96.9 against 88 on llama-3.1_8b.

On Arithmetic[†], the depth mean puts Ψ ahead on gemma-2_2b (37.7 against 31) and close behind DAS on llama-3.1_8b (52.5 against 54). On RAVEL (Appendix A.7, Table 3), Ψ ties or leads every published method in all measured models and variables. Lastly, the random floor is informative:

[†]Versions of Mueller et al. (2025) disagree on the published entries. We quote the latest arXiv version, which is the one we could reproduce from the authors' released code, and which differs from the proceedings only in this column. The official leaderboard carries no Arithmetic entry at the time of writing.

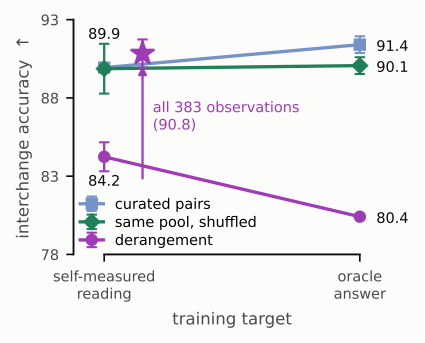

Figure 2: **What a counterfactual dataset buys.** MCQA X_{Order}, gemma-2_2b, layer 17; mean and standard deviation over three seeds. The two ingredients of counterfactual supervision are varied independently: how base and source are paired (color) and what the intervened model is trained towards (horizontal axis). The oracle answer helps the curated arm and hurts the deranged one, so an oracle label is worth having together with the oracle's pairing. The star trains the deranged arm on all 383 eligible observations, which a derangement can use because it needs no pair list.

at rank 16 it scores 33.3 at every MCQA X_{Order} site, and at rank 1152 it scores 19.3 on O_{Answer}, below chance; on RAVEL it sits above every published baseline in most cells, because most test pairs expect no change. On those that do, Ψ produces the changed answer 26–50% of the time and a random subspace never does.

Supervision ablation. A counterfactual dataset supplies two things: a curated pairing of each base prompt with a source, and the oracle answer that pair should produce. We focused on the MCQA task and the gemma-2_2b model, and varied the two ingredients independently at one site (layer 17), holding rank, optimizer and budget fixed, and restricting the unpaired arms to the 108 observations the benchmark's pair list uses as bases so that all four cells see the same prompts. Holding the pair list's own rows, its base multiplicities and its 544 optimizer updates, and permuting only which source accompanies which base, costs 0.07 points (89.94 against 89.87). The coupling a counterfactual dataset supplies is therefore worth almost nothing on its own. What it does supply is a donor pool: drawing donors from the same rows instead of from the pair list's distinct sources costs 5.63 points (84.24). A derangement needs no pair list and can therefore draw on all 383 eligible observations, where the curated arms are confined to the 108 the pair list uses as bases; doing so reaches 90.84 ± 0.9 against 91.40 ± 0.5 for the fully supervised cell. We conclude that a larger donor pool can replace explicit pairing supervision.

3.2 Controlling a variable without a donor

Correct localization does not by itself guarantee control (Galeone et al., 2026), and a carrier trained only to transport readings between prompts need not be writable to a reading that no prompt exhibits. We test this by training the carrier and the controller jointly, through the full objective of Eq. (10), at two sites in gemma-2_2b: MCQA X_{Order} at layer 17, where the declared library has four entries, and Arithmetic X_{Carry} at layer 21, where it has two. A reading q encodes a probability distribution over the declared library $\mathcal{O}$, and geometrically it is a point in the corresponding probability simplex. Its corners are the requests $q^\star$ with an unambiguous right answer, i.e. conditions for which the model should *answer as though the variable took value* v, while generic requests q^* are points in the interior of the simplex. Note that for a generic request q^* there is typically no prompt in the data that exhibits such a pattern and hence no donor that could provide it. During training, half the requests are readings other prompts exhibit and half are uniform draws from the simplex; we evaluate on held-out corner requests, where compliant steering behavior is well-defined. We report what the subject emits under unrestricted decoding (Figure 3a,b): requests are realized 94.8% of the time at the MCQA site and $63.7 \pm 2.6\%$ at the arithmetic one, above the donor interchange at both, which on arithmetic reaches 50.8%, at the 50% chance floor. Hollow dots give the declared-library rule the objective optimizes: the two agree to 2.3 points at MCQA but to 29.6 on arithmetic, where Ψ_{full} keeps the mass that CONDITIONAL gives up and overtakes it at 75.0% (Appendix A.4). The same controller inside a rank-matched DAS, principal or random subspace reaches 81.7%, 83.9% and 34.2%, and DAS loses 5.0 points between the two rules against our 2.3. Accuracy, however, only asks which answer wins, which fails to capture the nuance of a generic request q^*. In panel (c) we thus report the total variation distance to the requested reading on the interior requests q^*, which no donor exhibits, where Ψ reaches 0.10 against 0.18 for an interchange and 0.44 for no edit. In Appendix A.6 we report what an edit costs the rest of the distribution: Ψ pays about what a donor interchange does, at higher effectiveness, though Eq. (10) contains no cost penalty.

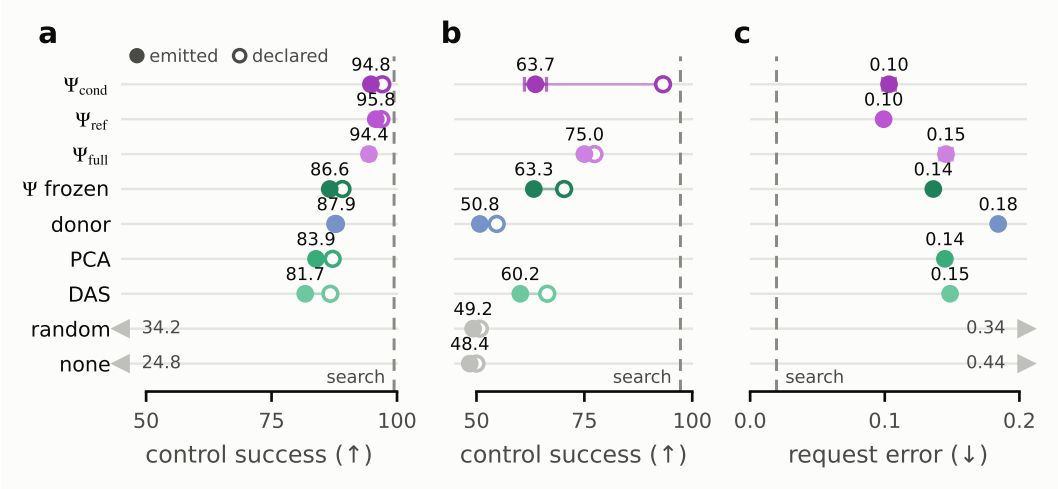

Figure 3: **A declared reading can be requested.** Rows are shared across panels: the first three train Ψ jointly under each mass mode, the next four attach the same controller to a frozen carrier. **(a, b)** Held-out corner requests realized at MCQA X_{Order} layer 17 and Arithmetic X_{Carry} layer 21 (gemma-2_2b); filled dots are greedy decoding and hollow ones the declared-library rule, on the same recipients. **(c)** Total variation to the target on interior requests. Seed whiskers (three seeds) mostly fall within the dots; the dashed line (*search*) optimizes each edit separately.

4 Discussion

A semantic program $\mathcal{P}$ replaces the executable high-level model $\mathbf{H}$ that causal-variable methods are supervised by, and we showed the same declaration interface can then be leveraged as the coordinate system in which a behavior can be requested, without explicit donor examples. In our experiments we showed that such an interface can successfully identify a low-rank causal subspace with Ψ showing competitive performance against available baselines (Table 1), often improving on the supervised alternatives. Our supervision ablation investigation (Figure 2) on the MCQA task corroborated these benchmark findings by revealing how the majority of the performance benefits actually stem from a net increase in the available donor pool size that a method can leverage during training, with explicit counterfactual pairing only contributing modest improvements. Finally, we showed that the same semantic program can be used to successfully steer the model's behavior without the need for an explicit donor prompt. We showed that joint localization and control training yields better control performance (Figure 3), both in terms of explicit control successes and request errors, than a controller built on top of a frozen localization-only carrier, which is in line with previous observations that explicitly differentiate the two concepts (Galeone et al., 2026).

Limitations. First, the semantic programs in this paper were easy to write. Their tasks have small, explicit answer sets, and for a more abstract variable it may be hard to name a library that is both semantically clean and broad enough to hold much of the model's probability. While we acknowledge this is a challenge, it is the same difficulty that plagues the full algorithm specification $\mathbf{H}$ required by other techniques, though in a milder form: a library and a rule are less to get right than an algorithm. Another limitation is that interchange accuracy by itself is not a guarantee of identification: a subspace that supports the declared transport τ need not be unique; moreover, a subspace can move the output through a pathway that has nothing to do with the variable it is supposed to name (Makelov et al., 2024). This is why we report localization and control separately and use only linear abstractions, since non-linear ones render causal abstraction trivial (Sutter et al., 2025). Finally, while our supervision is weaker than a full counterfactual dataset, it is not absent: gold answers are still used to filter the training pool, as they are for the published baselines (Mueller et al., 2025); such filtering increases the quality of the deranged pairs by avoiding prompts for which the model is unreliable. Future experiments can nevertheless quantify the impact of filtering on downstream localization performance, and whether it affects the supervised alternatives and Ψ asymmetrically.

AI USE STATEMENT

In this work we have used generative AI tools to refine hypotheses, provide feedback on experiments and assist the experiment lifecycle (e.g. submitting cluster jobs, writing scripts to collect results). We have not used AI tools to design the research, write the core Ψ algorithm, support qualitative and thematic data analysis, interpret results, generate synthetic datasets or assist with translation; the remaining tasks (e.g. theoretical proofs) are not applicable to this work. Additionally, we used generative AI tools to create or modify scientific figures, draft sections of the manuscript, write supporting code (e.g. tests, evaluation scripts), and summarize existing literature and identify gaps. We have reviewed all AI-assisted work, inspecting the code and writing and editing the full manuscript ourselves. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The source code and the run manifest can be found in the supplementary material and will be publicly released upon acceptance. Additionally, Appendix A.3 gives the complete semantic program for every task, which together with the objective of Section 2.2 and the public MIB dataset (Mueller et al., 2025) is everything an implementation needs to reproduce our results.

REFERENCES

Aryaman Arora, Dan Jurafsky, and Christopher Potts. CausalGym: Benchmarking causal interpretability methods on linguistic tasks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14638–14663, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.785. URL https://aclanthology.org/2024.acl-long.785/.

Yuntai Bao, Xuhong Zhang, Jintao Chen, Ge Su, Yuxiang Cai, Hao Peng, Sun Bing, Haiqin Weng, Liu Yan, and Jianwei Yin. Faithful bi-directional model steering via distribution matching and distributed interchange interventions. In *International Conference on Learning Representations*, 2026.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022. doi: 10.1162/coli_a_00422. URL https://aclanthology.org/2022.cl-1.7/.

Usha Bhalla, Suraj Srinivas, Asma Ghandeharioun, and Himabindu Lakkaraju. Towards unifying interpretability and control: Evaluation via intervention. *arXiv preprint arXiv:2411.04430*, 2025.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *International Conference on Learning Representations*, 2024.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, et al. Toy models of superposition. *arXiv preprint arXiv:2209.10652*, 2022.

Cosimo Galeone, Anna Ettorre, Minsu Park, Giuseppe Ettorre, and Daniele Ligorio. Perfect detection, failed control: The geometry of knowing vs. steering in language models. *arXiv preprint arXiv:2606.24952*, 2026.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. In *International Conference on Learning Representations*, 2025.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal abstractions of neural networks. In *Advances in Neural Information Processing Systems*, 2021.

Atticus Geiger, Zhengxuan Wu, Hanson Lu, Josh Rozner, Elisa Kreiss, Thomas Icard, Noah D. Goodman, and Christopher Potts. Inducing causal structure for interpretable neural networks. In *International Conference on Machine Learning*, 2022.

Atticus Geiger, Zhengxuan Wu, Christopher Potts, Thomas Icard, and Noah D. Goodman. Finding alignments between interpretable causal variables and distributed neural representations. In *Causal Learning and Reasoning (CLeaR)*, 2024.

Atticus Geiger, Duligur Ibeling, Amir Zur, Maheep Chaudhary, Sonakshi Chauhan, Jing Huang, Aryaman Arora, Zhengxuan Wu, Noah Goodman, Christopher Potts, and Thomas Icard. Causal abstraction: A theoretical foundation for mechanistic interpretability. *Journal of Machine Learning Research*, 26(83):1–64, 2025. URL https://jmlr.org/papers/volume26/23-0058/23-0058.html.

Jing Huang, Zhengxuan Wu, Christopher Potts, Mor Geva, and Atticus Geiger. RAVEL: Evaluating interpretability methods on disentangling language model representations. In *Annual Meeting of the Association for Computational Linguistics*, 2024.

Chonghe Jiang, Bao Nguyen, Anthony Man-Cho So, and Viet Anh Nguyen. Probe-free low-rank activation intervention. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 2812–2824, 2025.

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

Aleksandar Makelov, Georg Lange, Atticus Geiger, and Neel Nanda. Is this the subspace you are looking for? An interpretability illusion for subspace activation patching. In *International Conference on Learning Representations*, 2024.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, 2022.

Julian Minder, Kevin Du, Niklas Stoehr, Giovanni Monea, Chris Wendler, Robert West, and Ryan Cotterell. Controllable context sensitivity and the knob behind it. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=Igm9bbkzHC.

Aaron Mueller, Jannik Brinkmann, Millicent Li, Samuel Marks, Koyena Pal, Nikhil Prakash, Can Rager, Aruna Sankaranarayanan, Arnab Sen Sharma, Jiuding Sun, Eric Todd, David Bau, and Yonatan Belinkov. The quest for the right mediator: Surveying mechanistic interpretability through the lens of causal mediation analysis. *arXiv preprint arXiv:2408.01416*, 2024.

Aaron Mueller, Atticus Geiger, Sarah Wiegreffe, Dana Arad, Iván Arcuschin, Adam Belfki, Yik Siu Chan, Jaden Fiotto-Kaufman, Tal Haklay, Michael Hanna, Jing Huang, Rohan Gupta, Yaniv Nikankin, Hadas Orgad, Nikhil Prakash, Anja Reusch, Aruna Sankaranarayanan, Shun Shao, Alessandro Stolfo, Martin Tutek, Amir Zur, David Bau, and Yonatan Belinkov. MIB: A mechanistic interpretability benchmark. In *International Conference on Machine Learning*, 2025.

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

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 39643–39666. PMLR, 2024. URL https://proceedings.mlr.press/v235/park24c.html.

Philip Quirke and Fazl Barez. Understanding addition in transformers. In *International Conference on Learning Representations*, 2024.

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

Pau Rodriguez, Michal Klein, Eleonora Gualdoni, Valentino Maiorca, Arno Blaas, Luca Zappella, Marco Cuturi, and Xavier Suau. LinEAS: End-to-end learning of activation steering with a distributional loss. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2503.10679.

Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, et al. Open problems in mechanistic interpretability. *arXiv preprint arXiv:2501.16496*, 2025.

Yingdong Shi, Ruiming Zhang, Changming Li, Zhiyu Yang, Kaixing Zhang, Jingyi Yu, and Kan Ren. UniSteer: Text-guided flow matching in activation space for versatile LLM steering. *arXiv preprint arXiv:2605.30076*, 2026.

Alessandro Stolfo, Yonatan Belinkov, and Mrinmaya Sachan. A mechanistic interpretation of arithmetic reasoning in language models using causal mediation analysis. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7035–7052, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.435. URL https://aclanthology.org/2023.emnlp-main.435/.

Jiuding Sun, Jing Huang, Sidharth Baskaran, Karel D'Oosterlinck, Christopher Potts, Michael Sklar, and Atticus Geiger. HyperDAS: Towards automating mechanistic interpretability with hypernetworks. In *International Conference on Learning Representations*, 2025.

Denis Sutter, Julian Minder, Thomas Hofmann, and Tiago Pimentel. The non-linear representation dilemma: Is causal abstraction enough for mechanistic interpretability? In *Advances in Neural Information Processing Systems*, 2025.

Daniel Tan, David Chanin, Aengus Lynch, Brooks Paige, Dimitrios Kanoulas, Adrià Garriga-Alonso, and Robert Kirk. Analysing the generalisation and reliability of steering vectors. In *Advances in Neural Information Processing Systems*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/fb3ad59a84799bfb8d700e56d19c231b-Abstract-Conference.html.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*, 2023.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. Investigating gender bias in language models using causal mediation analysis. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/92650b2e92217715fe312e6fa7b90d82-Abstract.html.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *International Conference on Learning Representations*, 2023.

Zhengxuan Wu, Atticus Geiger, Thomas Icard, Christopher Potts, and Noah D. Goodman. Interpretability at scale: Identifying causal mechanisms in Alpaca. In *Advances in Neural Information Processing Systems*, 2023.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. ReFT: Representation finetuning for language models. In *Advances in Neural Information Processing Systems*, 2024.

Zhengxuan Wu, Aryaman Arora, Atticus Geiger, Zheng Wang, Jing Huang, Dan Jurafsky, Christopher D Manning, and Christopher Potts. AxBench: Steering LLMs? Even simple baselines outperform sparse autoencoders. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 67035–67080. PMLR, 2025. URL https://proceedings.mlr.press/v267/wu25a.html.

Daniel Wurgaft, Can Rager, Matthew Kowal, Vasudev Shyam, Sheridan Feucht, Usha Bhalla, Tal Haklay, Eric Bigelow, Raphael Sarfati, Thomas McGrath, et al. Manifold steering reveals the shared geometry of neural network representation and behavior. *arXiv preprint arXiv:2605.05115*, 2026.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to AI transparency. *arXiv preprint arXiv:2310.01405*, 2023.

A APPENDIX

A.1 ADDITIONAL RELATED WORK

The distributed alignment lineage. DAS learns a rotation whose coordinate block aligns with a high-level variable (Geiger et al., 2024), Boundless DAS learns the subspace boundary as well so the rank need not be fixed by search (Wu et al., 2023), MDAS optimizes attribute isolation alongside the alignment on RAVEL (Huang et al., 2024), and HyperDAS replaces the brute-force search over feature locations with a hypernetwork (Sun et al., 2025). All four take their training target from an executable high-level model, which is the input a semantic program replaces. HyperDAS in particular automates where to look; the declaration changes what the investigator has to write down, and the two are independent.

Writing a behavior into activations. Representation engineering (Zou et al., 2023) and activation addition (Turner et al., 2023; Rimsky et al., 2024) add to the residual stream a direction fitted from contrastive prompts, and inference-time intervention does the same along a truth-related direction (Li et al., 2023). Recent work learns the write operator instead of fitting it. CDAS matches the intervened output distribution against the model's own distribution on concept-eliciting prompts and clamps a rank-1 coordinate at inference (Bao et al., 2026); a single causally identified direction acts as a context-sensitivity knob across a model family (Minder et al., 2025); LinEAS fits per-layer affine maps against a distributional loss over activation populations (Rodriguez et al., 2025); a probe-free low-rank map pushes an activation towards a manifold of desirable content (Jiang et al., 2025); and UniSteer conditions a flow in activation space on a natural-language description of the wanted behavior (Shi et al., 2026). Each of weak supervision, model-relative targets, donor-free inference and amortization over a space of requests is therefore established on its own. What a semantic program adds is a request that is a distribution over a declared answer set indexed by a causal variable's values, issued to a subspace that interchange interventions have independently validated.

The write operator itself is LoReFT's, i.e. project into a low-rank subspace, compute a target there, add the residual back (Wu et al., 2024). Ours takes the requested reading as an argument, so one training run covers the whole simplex rather than one behavior.

Localization and control are different measurements. Detection directions and steering directions are close to orthogonal (Galeone et al., 2026), and the methods that win at concept detection and at steering are not the same ones (Wu et al., 2025). The geometry says why: the directions that read a concept off an activation and the directions that change it are related by the causal inner product and not by the Euclidean one (Park et al., 2024). Steering vectors are in addition variable across inputs and brittle to reasonable changes of prompt (Tan et al., 2024), and a subspace intervention can move the output through a dormant parallel pathway while the feature it names does nothing (Makelov et al., 2024). Intervention success as an evaluation criterion is itself established (Bhalla et al., 2025); we use it as a second measurement on one identified carrier rather than as a faithfulness criterion in its own right.

A.2 SUPERVISION ACCOUNTING

The row order runs from the most to the least that a method asks of the investigator before training begins. Every entry is an input, so a shorter row is a cheaper method to apply, and Section 3.1 measures what the entries in the DAS row are worth in accuracy.

A.3 THE SEMANTIC PROGRAM FOR EACH TASK

Everything the investigator writes for a task is below. Each program is an outcome library $\mathcal{O}$ and a transport rule $\mathcal{T}$, fixed before training and shared by every run on that task. Where a rule needs the source's reading and the source prompt does not supply one, we also give the query that measures it.

MCQA and ARC (Easy). Both are multiple-choice, and both variables use the full-change rule $\boldsymbol{\tau} = q(\mathbf{s})$. They differ in what the library names and in how wide it is. For X_{Order}, the position of the

Table 2: Supervision accounting. Each column is an input the investigator must provide before training begins, and the entries say *what* that input is: the methods differ in kind, not only in quantity. The DAS lineage (Geiger et al., 2024; Wu et al., 2023; Huang et al., 2024) shares one row because it asks for one thing. PSI replaces the algorithmic specification with a declaration in the answer space and the oracle counterfactual with a measurement; gold answers still enter, as the correctness filter on the training pool that the published baselines also apply. Remaining rows: HyperDAS (Sun et al., 2025), CDAS (Bao et al., 2026), LoReFT (Wu et al., 2024).

Method	High-level model	Intervention target	Pairing	Gold labels
DAS, Boundless DAS, MDAS	algorithm	oracle answer	curated	target, filter
HyperDAS	algorithm, text	oracle answer	curated	target, filter
CDAS	concept prompts	own distribution	contrast sets	set membership
LoReFT	none	task label	none	training target
Ψ (ours)	library, transport	own reading	derangement	filter

correct option, the library is the prompt's own choice labels in the order that prompt lists them, each with its leading space, indexed by option position; an interchange that moves the variable moves the model from one label to another. For O_{Answer}, the answer symbol, the library is the twenty-six leading-space letters $(\text{␣A}, \ldots, \text{␣Z})$, identical for every prompt. The two variables therefore do not share a library: one has four entries drawn from the prompt, the other twenty-six fixed ones, which is what lets a single benchmark task carry two variables at different widths.

Arithmetic (+). The variable is the carry out of the units digits, so the library has two entries: the sum with the carry propagated and the sum without it. For `47 + 28 =` these are `75` and `65`. Both are computed from the operands the prompt states and not from the answer, but writing that library does take knowing what a carry does to a sum. The program supplies the mapping from this variable's values to the strings they produce, and nothing else about how the subject computes them. The rule is again the full change. The carry is a natural variable to declare: it is the intermediate the benchmark itself targets (Mueller et al., 2025), transformers trained on addition compute it explicitly (Quirke & Barez, 2024), and causal mediation analysis has localized where language models route the operands of such sums (Stolfo et al., 2023).

RAVEL. The library is built once from the benchmark's entity file: every value any city takes on any of the three attributes, reduced to its first token under the subject's tokenizer and deduplicated, so a single library of country, continent and language names serves all three variables. Entries are therefore first tokens rather than complete attribute names, and two values sharing a first token are one entry. The rule is gated, $\boldsymbol{\tau} = \gamma(\mathbf{b})\, q(\mathbf{s}) + (1-\gamma(\mathbf{b}))\, q(\mathbf{b})$, with $\gamma(\mathbf{b}) = 1$ when the base prompt queries the attribute being intervened on and $\gamma(\mathbf{b}) = 0$ otherwise. That single bit is the whole of what RAVEL's isolation requirement asks for: change the answer when the question is about the attribute you moved, leave it alone when it is not.

RAVEL is also the one task where the source's reading cannot be taken from the source prompt. It has two prompt families: JSON-style continuations such as `{"city": "Nagoya", "language": "`, which name an attribute, and free text such as `Spartak Trnava won 3--1 on aggregate`, which names none. The program therefore declares the query that measures the source's value,

```
Q: What is the {attribute} of {entity}?
A:
```

evaluated on the frozen subject with `{attribute}` the one being intervened on. This is the declared query of Eq. (5): one extra forward pass per source observation, and the only place a program in this paper reaches beyond the prompts it was given.

IOI. We consider the IOI task an extension rather than an instance of the interface, and we separate it because its transport rule is not measured from the subject, due to the particular implementation of this task in the MIB benchmark itself. The benchmark scores the task as a signed logit margin between the indirect object and the subject name, and supplies a linear model of that margin, $\text{bias} +$

$\beta_{\text{tok}} s_{\text{tok}} + \beta_{\text{pos}} s_{\text{pos}}$, with fixed coefficients and each signal $+1$ when that variable still matches the base and -1 when it does not. We build the transport table by evaluating that model under the isolated intervention and passing the resulting margin through a sigmoid, which gives the probability the indirect object should carry; the library is the two names. The numbers in an IOI target therefore come from the benchmark's high-level model and from the gold identity of the base's indirect object, not from the frozen subject. Every claim in this paper about targets measured from the subject covers MCQA, ARC, arithmetic and RAVEL; it does not cover IOI, and Appendix A.7 reports that task on those terms.

Unused supervision. With the exception of IOI above, no program names a correct answer, and none names which library entry a given prompt should elicit. Gold answers enter the other four tasks outside the program, as the filter that keeps only observations the subject already answers correctly, and through the benchmark's own choice of cut position, which is located by the correct answer symbol. Both are conventions the published baselines also follow, and they are the reason we describe the supervision as weaker (rather than completely absent).

A.4 Two decision rules

The benchmark's evaluator decodes from the intervened model over the whole vocabulary and asks whether the expected answer appears. Our training objective works instead on the declared library, so we also audit with the rule the objective optimizes: restricted to the library, does the intervened model prefer the entry the transport rule asked for? Comparing the two tells us whether a score reflects the variable moving or the decode landing elsewhere.

The two agree closely wherever the declared answers are single tokens. They come apart on arithmetic, where an answer spans three tokens: at layer 19 the carrier scores 46.3 under the benchmark's rule and 71.3 under ours. The carry does move, and the rollout that follows it often lands on a number outside the declared pair. Every localization number in the main text uses the benchmark's rule; Section 3.2 reports both.

A.5 The mass term

Equation (7) splits the training signal into the shape of the reading and the probability mass the library holds, and Section 2.2 leaves open what to ask of the second term. We measured all three options at one site over three seeds to choose between them. Dropping the term (CONDITIONAL), aiming it at the mass the unintervened prompt already shows (REFERENCE), and maximizing it (FULL) give 90.84 ± 0.9, 91.09 ± 1.0 and 89.02 ± 0.0 on MCQA X_{Order}. FULL is the only mode that pays to keep probability on the declared answers, which is what should stop an undeclared token winning a decode, and it scores lowest here, so we use CONDITIONAL for every task except Arithmetic. There the answers span several tokens, and preserving the mass on the digits benefits greedy decoding: at the gemma-2_2b X_{Carry} site (layer 20, three seeds) FULL reaches 45.94 ± 0.1 against 45.31 ± 0.2 for CONDITIONAL under the benchmark's rule, even though it trails under the declared one (68.9 against 73.1). The control experiments show the same ordering under greedy decoding (Figure 3b and below), so the Arithmetic localization runs use FULL.

The same choice is equally important for control, where the two terms of the decomposition pull against each other. At the arithmetic site Ψ_{full} keeps 0.661 of the declared mass after the edit against 0.291 for Ψ_{cond}, and pays 15.6 points of control success under the declared rule for it (77.3% against $93.2 \pm 4.3\%$), and buys it back under free decoding, where the order reverses (75.0% against $63.7 \pm 2.6\%$): the mass CONDITIONAL gives up is mass the undeclared tail takes. Asking the model to keep talking about the declared answers and asking it to change which one it gives are different requests, and the decomposition is where a program states which of the two it wants.

A.6 What the control term costs

Carrier cost. The two terms of Eq. (10) share the subspace A, so it is reasonable to expect them to compete: a subspace shaped to be writable need not be the best carrier under transplantation. We measured the effect at the MCQA X_{Order} site of gemma-2_2b, training three seeds either for localization alone ($\lambda = 0$ in Eq. (10)) or jointly ($\lambda = 1$). Seed for seed, turning the control term

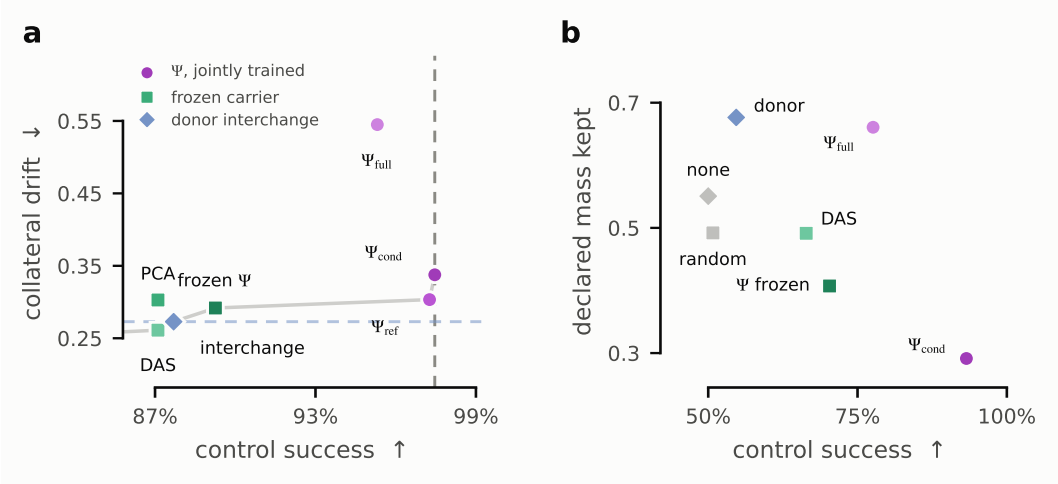

Figure 4: Cost against success for the arms of Figure 3, seed 11, the same held-out recipients and the same corner requests, so the two axes are comparable point by point. **(a)** MCQA X_{Order}: collateral drift is the total-variation distance between the undeclared tails of the output distribution before and after the edit, each renormalized, and the dashed line marks what a donor interchange costs at the same site. **(b)** Arithmetic answers span several tokens, so the undeclared tail is ill defined there and the cost axis is the probability mass the declared library retains.

on lowers interchange accuracy by 1.9 points and raises control success by 7.9. The cost is small but consistent, and it is why the benchmark evaluations of Section 3.1 use $\lambda = 0$ while the control experiments use $\lambda = 1$.

The joint carrier is also trained for longer than the frozen ones it is compared against, so we checked whether its advantage is the training budget. Attaching the same controller to donor-only carriers trained for 8, 32 and 64 epochs gives (under the declared rule) 89.6%, 90.4% and 90.2% of requests realized, against 97.5% for the jointly trained carrier on the same recipients. Four to eight times the carrier budget is worth under a point and then saturates, so we attribute the gap to full joint training.

Collateral drift. The control term also competes with the rest of the model's behavior and altering the model's preferences within the declared library $\mathcal{O}$ might come at the expense of the overall probability mass outside the library, i.e. on $\mathcal{O}_\perp$. Figure 4 plots the measured controlled accuracies for various methods against a measure of collateral control cost, revealing a Pareto-like tradeoff between control efficacy and intervention cost.

We quantified the collateral cost differently between the MCQA task (Figure 4a) and the arithmetic task (Figure 4b), given that what can be measured differs by site. MCQA's declared outcomes are disjoint single tokens, so the undeclared tail is well defined and drift can be read over the whole vocabulary; we thus quantified the cost as the total variation between the tail of the distribution before the edit and the tail of the distribution after the edit. Arithmetic's answers are three tokens long, and there the available axis is how much probability the declared library still holds after the edit.

A.7 RAVEL AND IOI

RAVEL. The RAVEL task (Huang et al., 2024) queries a language model about the continent, country and language of a city, and asks a localization method to change one attribute while leaving the others alone. A base prompt queries one attribute at a time, so an intervention on an attribute the prompt is not asking about is supposed to change nothing: only 17 to 38% of the public test pairs expect the answer to move at all. The benchmark's accuracy averages the two kinds of pair together, and the no-op majority dominates it. This is why the random subspace control matters more here than anywhere else in the paper: a subspace that does nothing answers most pairs correctly, and

Table 3: **RAVEL and IOI.** Top, RAVEL: conventions as in Table 1, interchange accuracy with the best layer in parentheses, higher better, except that the random subspace competes for bold, since it scores above every published baseline in most cells. Bottom, IOI on gpt2-small, whose site is a fixed set of attention heads (three for S_{Pos}, one for S_{Tok}) rather than a swept layer. The benchmark scores it as a mean-squared error between the causal model's logit difference and the subject's, so lower is better there; we add the interchange accuracy of the same runs, which the publication does not report, and our own run of DAS through the same evaluator, because the published S_{Tok} entry is one we could not reproduce. Accuracy is not bolded where doing nothing already scores highest. A dash marks a number the source does not provide.

	RAVEL					
	gemma-2_2b			llama-3.1_8b		
Method	A_{Cont}	A_{Ctry}	A_{Lang}	A_{Cont}	A_{Ctry}	A_{Lang}
Full Vector	48 (62)	49 (57)	45 (56)	53 (62)	47 (53)	47 (57)
Random	85 (86)	61 (62)	62 (63)	**82** (83)	64 (65)	65 (65)
DBM	66 (71)	53 (65)	54 (58)	68 (80)	53 (59)	58 (64)
+ PCA	63 (70)	47 (53)	50 (56)	62 (74)	48 (54)	53 (57)
+ SAE	64 (72)	49 (56)	53 (59)	64 (72)	50 (57)	55 (57)
DAS	75 (85)	57 (**67**)	62 (**70**)	75 (83)	58 (64)	63 (70)
Ψ (ours)	**86** (**89**)	**63** (**67**)	**65** (**70**)	81 (**84**)	**66** (**72**)	**66** (**71**)

	IOI			
	MSE ↓		IIA (%) ↑	
Method	S_{Pos}	S_{Tok}	S_{Pos}	S_{Tok}
No Change	16.19	5.39	33.3	100.0
Random	5.87	3.69	40.5	99.4
Full Vector	2.45	2.82	—	—
DBM	2.22	2.35	—	—
+ PCA	2.24	2.33	—	—
DAS	**2.20**	**2.08**	—	—
DAS (reproduced)	2.27	2.93	79.0	98.8
Ψ (ours)	2.29	3.26	**80.1**	99.7

Table 3 shows a random subspace of the matched rank scoring at or above every published baseline on the depth mean in all six columns.

Splitting the same audit by the gate separates the two questions. On the pairs that do expect a change, Ψ produces the right changed answer between 26 and 50% of the time: for instance 47.3% at the gemma-2_2b A_{Lang} site the table reports, whereas a random subspace of the same rank produces it 0.0% of the time at every one of the six sites. The aggregate conceals that separation completely. We report the aggregate in Table 3 because it is the benchmark's own metric. Ψ leads every published method on it, but its margin over the random subspace is small in every column, and on the llama-3.1_8b A_{Cont} depth mean the random subspace scores higher, which is why the table bolds it there; the split by the gate is the more informative comparison.

IOI. This is the one task whose transport rule is supplied by the benchmark's high-level model rather than measured from the subject, as Appendix A.3 sets out, and we report it on those terms. Its subject is gpt2-small. Indirect object identification (Wang et al., 2023) asks a model to complete a sentence such as "`When Mary and John went to the store, John gave a drink to`" with the name that appeared once rather than twice. MIB derives two variables from it: S_{Pos}, the position of the repeated subject, and S_{Tok}, its token identity. This task is scored differently from every other one in the benchmark. Rather than asking whether the intervened model produces the expected answer, it measures:

$$\mathbb{E}_{\iota}\big[\,\big(\Delta_{\phi}(\iota) - \Delta_{\mathbf{H}}(\iota)\big)^{2}\,\big], \qquad \Delta = \mathrm{logit}(\texttt{IO}) - \mathrm{logit}(\texttt{S}),$$

i.e., the mean-squared error between the logit difference the intervened subject produces and the one the high-level model $\mathbf{H}$ prescribes; lower is better. The abstraction site is also fixed: the alignment is carried by designated attention heads previously identified by Wang et al. (2023) (three for S_{Pos} and one for S_{Tok}) rather than by a swept layer.

Table 3 (bottom) reports our results. Under the objective used everywhere else in this paper, Ψ lags behind the official DAS results and the other baselines: it reaches 2.29 on S_{Pos} and 3.26 on S_{Tok}. We also ran our own reproduction of the DAS method, which reaches 2.27 on S_{Pos}, close to the official 2.20, but 2.93 on S_{Tok}, substantially higher than the official 2.08.

Two readings of those numbers are worth separating. The first is that the two variables are not equally decisive, and both metrics say so. The MSE has no natural zero, so it is read between its two nulls. On S_{Pos} doing nothing costs 16.19, a random subspace of the matched rank 5.87, and every trained method between 2.20 and 2.29. On S_{Tok} the same three anchors are 5.39, 3.69 and 2.08 to 3.26: the whole scale there is narrower than the stretch a random draw already covers on S_{Pos}. Scored as an interchange accuracy on the same audits, doing nothing reaches 33.3% and 100.0% respectively: moving the subject's position changes which name the model should emit, and moving its token identity does not. On the variable where the answer does move, Ψ reaches 80.1% against 79.0% for our DAS reproduction, compared with 77.2% for a whole-vector interchange at the same heads and 40.5% for a random subspace. On the other, every method except the whole-vector interchange, doing nothing included, sits within a point of the ceiling, and only the logit difference can be measured at all, which is precisely the reason the benchmark authors opted for the MSE metric on this task.

The second concerns what the objective aims at. A reading over two single-token names expresses the benchmark's margin exactly, since $\log(q_{\mathtt{IO}}/q_{\mathtt{S}}) = \mathrm{logit}(\mathtt{IO}) - \mathrm{logit}(\mathtt{S})$, and the transport table we build for IOI is precisely such a margin passed through a sigmoid. What differs is the loss and where its numbers come from: we minimize a cross-entropy against that target while the benchmark reports a squared error on the margin itself, so the two weight deviations differently, and the target is supplied by the benchmark's linear model rather than measured from the subject (Appendix A.3). We therefore read the distance between 3.26 and the published 2.08 as the cost of optimizing a different quantity under a supervision the rest of this paper does not use. The distance between the same published entry and our own reproduction of the method that produced it, 2.93, should be read alongside it.

A.8 Depthwise localization accuracy

In this section we report for completeness the full depthwise intervention accuracy profiles for the different MIB tasks. The official benchmark scores reported in Table 1 in fact only report two numbers per cell, i.e. mean and maximum. Figure 5 gives the profile they are computed from, which illustrates patterns that the aggregate score hides. On MCQA and ARC the token positions specialize with depth: an early layer is read best at the answer symbol and a late one at the last token, and the envelope is high across most of the network because one position or another carries the variable at each depth. On RAVEL the envelope tracks the random subspace along its whole length.

A.9 Environment and compute

All experiments are implemented in PyTorch (2.12) and evaluated by the benchmark's own evaluator at a pinned commit, in its own pinned environment, so our scores and the published baselines are produced by the same code path.

The localization results sweep every layer of each subject against the token positions the benchmark defines for that task, three for MCQA and two for ARC, arithmetic and RAVEL, which is 52 to 96 training runs per cell and 1032 in total. Training is one run per site, for 32 epochs on MCQA, two on ARC and one on arithmetic and RAVEL, matching the budgets under which the baselines were tuned. These ran on H100 cards; the random floors, which need no training, and the control and supervision experiments ran on three RTX A4000 cards. Cumulative compute for the project, including preliminary experiments that do not appear here, amounted to approximately 200 GPU-hours.

A.10 TRAINING HYPERPARAMETERS

Every run in this paper trains the carrier with AdamW (weight decay 10^{-4}, gradient norm clipped at 1) and keeps A orthonormal by taking the QR factor of an unconstrained matrix. The frozen subject runs in half precision, except gpt2-small, which runs in full precision. Single-seed results use seed 11 and three-seed results use seeds 11 to 13. Each batch draws its sources by a fresh random derangement among prompts of the same transport group, and the carrier rank is the official DAS rank of each cell (Section 3). Table 4 lists what differs between experiments; every value is read from the frozen configuration of the corresponding runs, which the supplementary material ships.

Table 4: Training settings. Localization runs use $\lambda = 0$ in Eq. (10). Control runs use $\lambda = 1$ and the gain controller of Eq. (9), trained with learning rate 0.1 against requests drawn half from other prompts' readings and half uniformly from the simplex, with edit budget $c = 104.4$ at the MCQA site and $c = 208.6$ at the Arithmetic site (Section 2.3). Where a row lists several batch sizes, they follow the order of its subjects.

Experiment	Subjects	Epochs	Batch	Learning rate	Mass mode
Localization, every layer and benchmark position (Tables 1 and 3)					
MCQA	gemma-2_2b, llama-3.1_8b, qwen-2.5_0.5b	32	16, 8, 32	3×10^{-4}	CONDITIONAL
ARC (Easy)	gemma-2_2b, llama-3.1_8b	2	16	3×10^{-4}	CONDITIONAL
Arithmetic	gemma-2_2b, llama-3.1_8b	1	8	10^{-3}	FULL
RAVEL	gemma-2_2b, llama-3.1_8b	1	32, 16	10^{-3}	CONDITIONAL
IOI, fixed heads	gpt2-small	2	32	10^{-3}	CONDITIONAL
Supervision ablation (Figure 2)					
MCQA X_{Order}, layer 17	gemma-2_2b	32	16	3×10^{-4}	CONDITIONAL
Control (Figure 3)					
MCQA X_{Order}, layer 17	gemma-2_2b	32	16	3×10^{-4}	all three
Arithmetic X_{Carry}, layer 21	gemma-2_2b	1	8	10^{-3}	CONDITIONAL, FULL

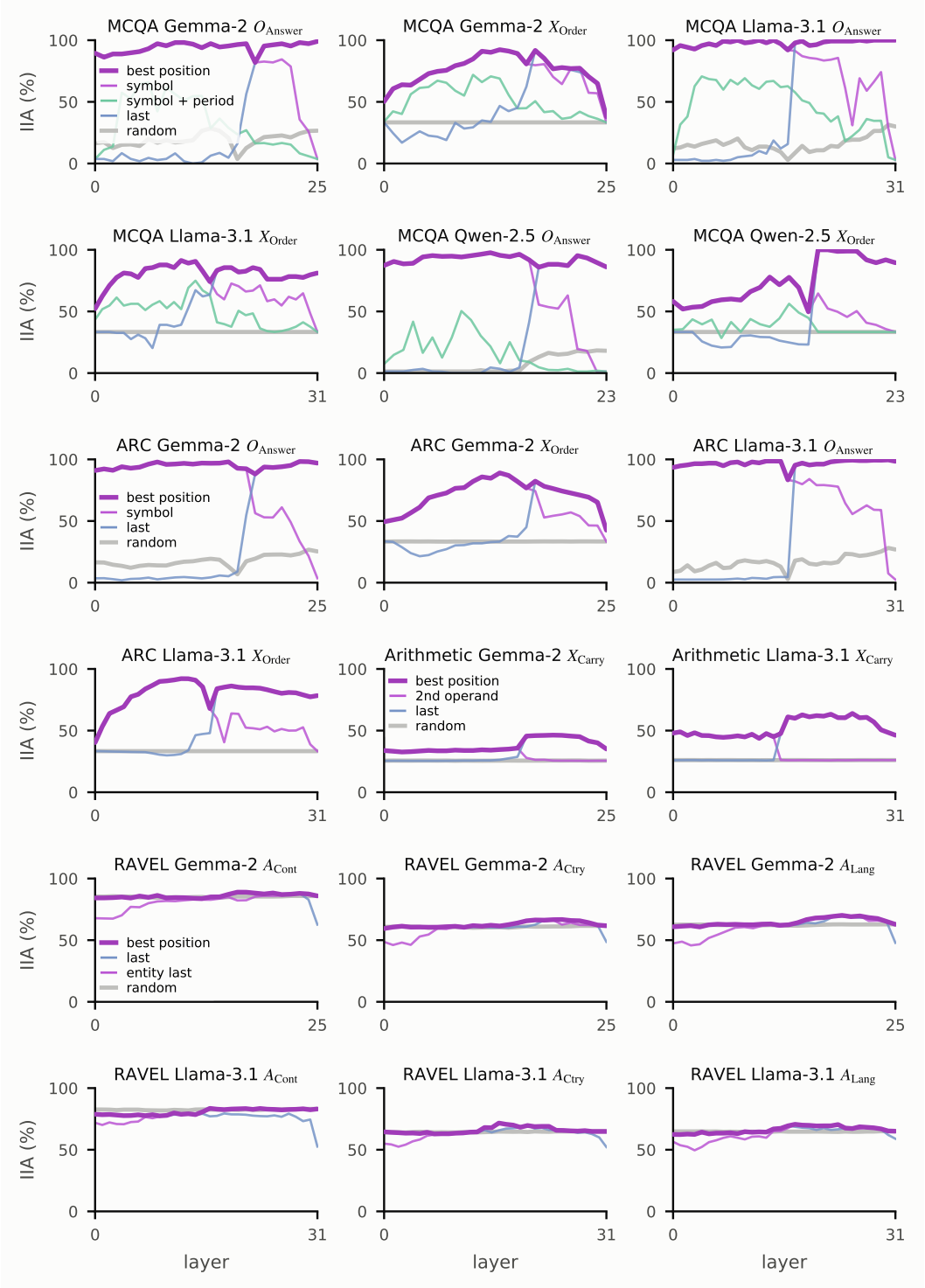

Figure 5: **Depthwise localization accuracy.** Interchange accuracy of Ψ at every layer of every cell, one thin line per token position and the thick line their maximum at each layer. The thick line is what Table 1 summarizes: its average over layers is the depth mean and its highest point is the best layer. The gray line is the random subspace of the same rank, read the same way, wherever we have measured it.