PLOT: PROGRESSIVE LOCALIZATION VIA OPTIMAL TRANSPORT IN NEURAL CAUSAL ABSTRACTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Causal abstraction offers a principled framework for mechanistic interpretability, aligning a high-level causal model with low-level neural computation through interchange intervention analysis. Finding such an alignment, however, often requires fitting and evaluating separate learned mappings across many candidate neural locations. We introduce PLOT, a gradient-free approach for joint correspondence discovery via a global matching of intervention effects. PLOT represents abstract and neural interventions by geometric signatures of their effects on the shared task output and fits an optimal transport coupling between the two collections. The coupling can be calibrated directly into an executable intervention handle or used progressively to select coarse parent sites for finer localization, reducing the cost of searching broad collections of neural sites. Experiments on hierarchical equality, binary addition, and multiple-choice question answering (MCQA) demonstrate fast and accurate direct handles without learning intervention rotations and show that progressive localization can improve accuracy under intervention-size constraints while reducing computational cost. PLOT can also guide gradient-based subspace learning, with PLOT-guided distributed alignment search (DAS) attaining accuracy comparable to full DAS at approximately $3.6\times$ and $20\times$ lower serial runtime on binary addition and MCQA, respectively. PLOT thus separates the search for candidate correspondences from optional subspace learning and provides a common framework for constructing and testing neural intervention handles for a given causal model.

1 INTRODUCTION

Causal abstractions of neural models are central to mechanistic interpretability. Given a trained neural network and a high-level model with interpretable latent variables, the goal is to determine whether the network realizes the abstract computation (Pearl, 2009; Geiger et al., 2025). Under this view, interpretation is not merely finding correlated neurons or salient features. Rather, an abstract variable is realized only if interventions on it can be matched by interventions at internal neural sites. This turns interpretability into a causal localization problem of finding a faithful correspondence between abstract variables and their high-dimensional, distributed neural realization.

Recent work has made this question operational through interchange interventions and distributed alignment search (DAS) (Geiger et al., 2022; 2024; Wu et al., 2023). DAS fixes a neural site, such as a layer, and learns a rotation under which swapping a selected subspace between source and base activations reproduces the abstract intervention. This makes DAS expressive and accurate (Mueller et al., 2025), but finding the relevant location can require fitting and evaluating alignments across many neural sites. Thus, DAS is powerful once a site has been proposed, while correspondence discovery across a broad candidate family remains a separate problem. We address this problem via a joint matching between the intervention effects of abstract variables and candidate neural sites.

1.1 CONTRIBUTIONS

We propose PLOT, a gradient-free approach for joint correspondence discovery via a global matching of intervention effects. Abstract and neural states may live in different spaces, but interventions on them can be compared through their effects on the common task output. PLOT records these effects across base/source pairs as geometric signatures and fits an optimal transport (OT) coupling between the abstract and neural signature collections. The coupling gives a global soft correspondence between

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

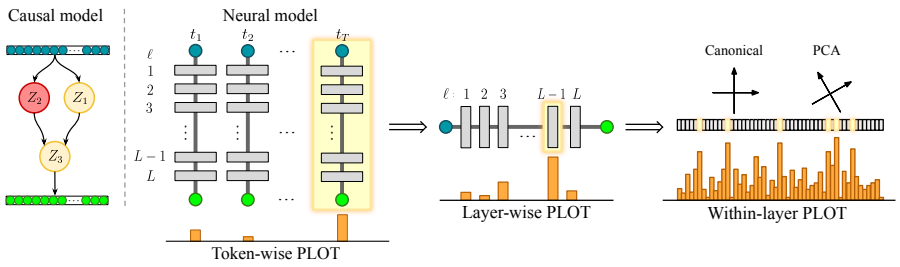


Figure 1: Progressive localization via optimal transport. The diagram follows one high-level variable, Z_2 in red, although each OT stage jointly matches all target variables to the neural site family. PLOT first localizes coarse sites, such as tokens/layers, then refines within them in native or PCA coordinates.

high-level variables and candidate neural sites that can be calibrated into executable handles, restrict the candidate family for subsequent localization, or guide gradient-based subspace learning.

The neural candidate family can range from individual neurons or coordinate blocks to PCA spans or entire activation vectors, and may include sites from multiple layers or timesteps. PLOT can match all such candidates jointly, allowing the calibrated handle to combine sites across neural locations. When the candidate family contains many irrelevant sites, we use one-sided unbalanced OT (UOT), which fixes the abstract marginal while relaxing the neural marginal so that irrelevant candidates need not absorb transport mass. PLOT can also exploit a natural hierarchy among the candidate sites through progressive localization. A coarse coupling first identifies promising parent sites, after which the matching is refined only within the selected locations. As illustrated in Figure 1, this can proceed from coarse sites such as tokens or layers to within-layer coordinate groups or PCA spans. The progressive procedure can improve the accuracy of compact handles while reducing localization cost or can be used to guide local subspace methods such as DAS.

Empirically, we study PLOT on three tasks of increasing localization complexity. On the hierarchical equality benchmark (HEQ) of Geiger et al. (2024), single-stage PLOT over MLP coordinates recovers accurate, compact handles without learned rotations or gradient-based optimization. On 4-bit binary addition with a GRUCell backbone, one- and two-stage PLOT recover accurate carry handles, with PCA outperforming native coordinates and two-stage localization improving accuracy and runtime. PLOT-guided DAS matches full-DAS accuracy at roughly $3.6\times$ lower runtime. Finally, on the multiple-choice question-answering (MCQA) benchmark (Mueller et al., 2025) with Gemma-2-2B (Gemma Team, 2024), we compare direct PLOT handles against masking- and patching-based baselines, as well as PLOT-guided DAS and full DAS. Two-stage PLOT improves over one-stage matching under matched handle-dimension limits while substantially reducing runtime, and its gradient-free PCA handles are within only 3.5 percentage points of full DAS accuracy. PLOT-guided DAS remains close to full DAS while reducing runtime by roughly $20\times$.

1.2 Literature Review

A broad line of mechanistic interpretability work seeks to localize internal structures that mediate specific model behaviors, spanning factual associations (Meng et al., 2022), learned features (Cunningham et al., 2023), and algorithmic circuits (Mueller et al., 2025; Polyakov et al., 2025). These methods are often geared toward feature discovery, model editing, or benchmark-specific analysis rather than a general causal correspondence criterion. The causal abstraction framework sharpens this target by asking whether an intervention on a hypothesized high-level variable can be reproduced by an internal intervention in the network (Geiger et al., 2025). Methods developed for this framework include interchange intervention training (IIT) (Geiger et al., 2022), DAS (Geiger et al., 2024), and subsequent methods that automate parts of the DAS search (Wu et al., 2023; Sun et al., 2025). PLOT instead constructs a joint correspondence by matching the output effect signatures of abstract variables and candidate neural sites. The resulting coupling can directly define intervention handles or be applied progressively to restrict subsequent localization and guide local subspace methods such as DAS. Related approaches based on PCA (Tigges et al., 2023; Marks & Tegmark, 2024), sparse autoencoders (SAEs) (Bricken et al., 2023; Huben et al., 2024), and masking (Chaudhary & Geiger, 2024; Mueller et al., 2025) provide candidate representations and additional localization baselines.

OT has also been used to align causal models across levels of abstraction. Most notably, causal OT of abstractions (COTA) (Felekis et al., 2024) formulates abstraction between causal models at different

granularities as a multimarginal OT problem (Gangbo & Święch, 1998). COTA enforces causal consistency through do-calculus constraints and intervention-aware transport costs. This setting, which aligns two explicit causal models, differs from PLOT, which uses OT to align a high-level causal model with the internal computation of a neural network. More broadly, our work is also related to causal OT and OT-based distances for causal models (Acciaio et al., 2020; Lassalle, 2018; Cheridito & Eckstein, 2025), though these works do not study neural causal abstraction.

2 Background and Preliminaries

2.1 Causal Abstraction of Neural Computation

Causal abstraction in mechanistic interpretability (Pearl, 2009; Geiger et al., 2025) compares an abstract causal model $\mathfrak{C} : \mathcal{X} \to \mathcal{Y}$, with interpretable variables $Z_1, \ldots, Z_m$, to a trained neural model $\mathfrak{N} : \mathcal{X} \to \mathcal{Y}$ that solves the same task. Formal notions such as exact transformations require consistency under corresponding interventions after mapping low-level states to high-level states (Rubenstein et al., 2017; Beckers & Halpern, 2019). Constructive abstractions additionally require this map to factor over disjoint groups of low-level variables. Here, we consider the neural localization setting of Geiger et al. (2021); Mueller et al. (2025), in which proposed correspondences yield intervention handles that are assessed through *interchange interventions*.

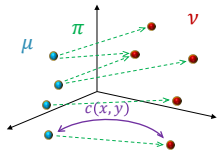
Abstract swap. Given a base/source pair of inputs $(x^b, x^s) \in \mathcal{X}^2$ and an abstract variable Z_i, the corresponding abstract counterfactual output is $y_i^{\text{abs_swap}}(x^b, x^s) := \mathfrak{C}\left(x^b; \text{do}\big(Z_i \leftarrow Z_i(x^s)\big)\right)$. This intervention keeps the base input fixed while replacing only the value of Z_i by its source value and recomputing the downstream variables. The operation is an interchange intervention, while its result is a base-specific counterfactual outcome.

Neural swap. A neural site s_j specifies a location and a collection of coordinates or features to intervene on, with values $s_j(x)$ on input x. A site may comprise a full layer activation vector, selected PCA directions or SAE features, a neuron subset selected by DBM (Chaudhary & Geiger, 2024), or a k-dimensional rotated subspace learned by DAS (Geiger et al., 2024). The analogous neural interchange output is $y_j^{\text{nn_swap}}(x^b, x^s) := \mathfrak{N}\left(x^b; \text{do}\big(s_j \leftarrow s_j(x^s)\big)\right)$. For coordinate and orthogonal subspace swaps, replacement leaves the complementary coordinates at that layer unchanged.

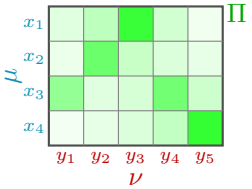
Correspondence criterion. For each abstract variable Z_i, an intervention handle H_i specifies which neural sites to modify and how. Writing $y_{H_i}^{\text{nn_swap}}(x^b, x^s)$ for its output, *interchange intervention accuracy* (IIA) is the fraction of evaluated base/source pairs satisfying $y_{H_i}^{\text{nn_swap}}(x^b, x^s) \sim y_i^{\text{abs_swap}}(x^b, x^s)$, where $\sim$ denotes task-dependent output agreement (Geiger et al., 2024; Mueller et al., 2025). A handle specifies an intervention rather than a map from neural state values to abstract values. High empirical IIA supports the proposed correspondence on the evaluated pairs, but does not by itself establish an exact or constructive abstraction.

2.2 Optimal Transport

OT compares two probability distributions by the minimum cost of moving mass from one to the other. Given $\mu \in \mathcal{P}(\mathcal{X})$, $\nu \in \mathcal{P}(\mathcal{Y})$, and a cost function $c : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, the Kantorovich OT problem is (Villani, 2003; Peyré & Cuturi, 2019) $\mathsf{OT}_c(\mu, \nu) := \inf_{\pi \in \Pi(\mu,\nu)} \int c(x, y)\, d\pi(x, y)$, where $\Pi(\mu, \nu) := \{\pi \in \mathcal{P}(\mathcal{X} \times \mathcal{Y}) : \pi(\cdot \times \mathcal{Y}) = \mu,\ \pi(\mathcal{X} \times \cdot) = \nu\}$ is the set of all couplings between μ and ν, each specifying how mass under μ is matched to mass under ν. The OT coupling can be viewed as a *soft correspondence*, since the mass associated with one support point of μ may be split across multiple support points of ν, with the coupling weights quantifying the association strength. We use the squared Euclidean cost $c(x, y) = \|x - y\|^2$; see Figure 2a. For discrete measures with m and n support points, the optimal coupling is a nonnegative matrix $\Pi \in \mathbb{R}_{\geq 0}^{m \times n}$, as illustrated in Figure 2b, whose entry $\Pi_{i,j}$ records the mass transported from support point i of μ to support point j of ν. In PLOT, the coupling matrix will provide the soft correspondence between abstract variables and candidate neural sites.



(a) A coupling $\pi \in \Pi(\mu, \nu)$ transports mass from μ to ν.



(b) Coupling matrix Π, representing the correspondence.

Figure 2: Optimal transport.

Entropic OT. For computation, we use entropically regularized OT

$$\mathsf{OT}_c^{\varepsilon}(\mu,\nu) = \inf_{\pi\in\Pi(\mu,\nu)} \int c\, d\pi + \varepsilon \mathsf{KL}(\pi \| \mu\otimes\nu), \tag{1}$$

where $\varepsilon > 0$. The added Kullback-Leibler (KL) divergence penalty makes the problem strictly convex and enables efficient computation via Sinkhorn iterations, with $O(mn)$ work per iteration for measures with m and n support points (Cuturi, 2013; Peyré & Cuturi, 2019).

Unbalanced OT. Standard OT enforces both marginals exactly, which can be too rigid when some candidate neural sites are irrelevant to the target variables. UOT relaxes the strict marginal constraints by penalizing, rather than forbidding, marginal mismatch (Chizat et al., 2018; Peyré & Cuturi, 2019):

$$\mathsf{OT}_c^{\varepsilon,\beta_1,\beta_2}(\mu,\nu) \coloneqq \inf_{\pi\in\mathcal{M}_+(\mathcal{X}\times\mathcal{Y})} \int c\, d\pi + \varepsilon \mathsf{KL}(\pi\|\mu\otimes\nu) + \beta_1 \mathsf{KL}(\pi_1\|\mu) + \beta_2 \mathsf{KL}(\pi_2\|\nu), \tag{2}$$

where π_1 and π_2 are the first and second marginals of π, which need not coincide exactly with μ and ν under the UOT framework. Large β_1, β_2 values recover balanced EOT, while finite values allow some mass to remain unmatched. In PLOT, when the neural candidate family is large and may contain irrelevant sites, we use the one-sided version that fixes the abstract marginal $\pi_1 = \mu$ and relaxes only the neural marginal π_2, with parameter $\beta = \beta_2$. This allows the coupling to assign little mass to irrelevant neural sites while retaining all abstract variables in the correspondence.

3 Progressive Localization via Optimal Transport

We now describe PLOT for localizing neural intervention handles associated with abstract variables $Z_1, \ldots, Z_m$, using the setting of Section 2.1. Assume the neural model has L layers, whose activation vectors, given x, are denoted by $h_\ell(x) \in \mathbb{R}^{d_\ell}$, $\ell = 1, \ldots, L$. We define candidate *neural sites* $s_1, \ldots, s_n$, each specifying an intervention subspace within a parent layer. A site may range from an entire activation vector to a coordinate group or PCA span, and the candidate family may span multiple layers. We denote the dimension of s_j by r_j, with activations $s_j(x) \in \mathbb{R}^{r_j}$ on input x.

Our method requires three datasets of base/source inputs: $\mathcal{D}_{\mathrm{ft}}, \mathcal{D}_{\mathrm{cal}}, \mathcal{D}_{\mathrm{te}}$, corresponding to fit, calibration, and test banks, which are used, respectively, to construct signatures, select localization/handle parameters, and evaluate the selected handles. We depart from the conventional train/validation/test terminology to better reflect the distinct roles these datasets play in our method. We first delineate a single PLOT stage and then discuss progressive localization.

3.1 Single PLOT Stage

We first describe the four steps for constructing and evaluating an intervention handle.

Step 1: Effect signatures. We use $\mathcal{D}_{\mathrm{ft}}$ to record the output effects of interchange interventions on each abstract variable $Z_1, \ldots, Z_m$ and neural site $s_1, \ldots, s_n$. These output effects are called *effect signatures*, which are constructed to lie in a common space and serve to align the models via OT.

Causal signatures: As in Section 2.1, for each pair $(x_t^b, x_t^s) \in \mathcal{D}_{\mathrm{ft}}$ and variable Z_i, write $y_{i,t}^{\mathrm{abs_swap}}$ for the abstract interchange output and $y_t^{\mathrm{abs}} = \mathfrak{C}(x_t^b)$ for the factual output. The *causal effect signature* is

$$\Delta_{i,t}^{\mathrm{abs}} \coloneqq d^{\mathrm{abs}}\big(y_{i,t}^{\mathrm{abs_swap}}, y_t^{\mathrm{abs}}\big), \qquad i = 1, \ldots, m, \quad t = 1, \ldots, T_{\mathrm{ft}},$$

where we typically take $d^{\mathrm{abs}}(y, y') = y - y'$ in the chosen numerical encoding of the outputs.

Neural signatures: Let $\ell(j) \in \{1, \ldots, L\}$ denote the parent layer of site s_j, so that s_j can be viewed as an r_j-dimensional intervention subspace of the layer activation space $\mathbb{R}^{d_{\ell(j)}}$. Let $W_j \in \mathbb{R}^{d_{\ell(j)} \times r_j}$ with $W_j^\intercal W_j = I_{r_j}$ have columns forming an orthonormal basis for this subspace. Thus, $W_j^\intercal$ maps the full layer activation $h_{\ell(j)}(x)$ to its coordinates within site s_j, while W_j maps changes in these coordinates back into the full layer activation space. We thus have $s_j(x) \coloneqq W_j^\intercal h_{\ell(j)}(x) \in \mathbb{R}^{r_j}$. In the canonical basis, the columns of W_j are canonical basis vectors selecting an individual neuron or coordinate block. In the PCA basis, they are PCA directions fitted to the layer's fit-bank activations.

Given (x_t^b, x_t^s), the neural swap/intervention at s_j replaces the base coordinates in this subspace by the corresponding source coordinates, while leaving the orthogonal complement unchanged:

$$h_{\ell(j)} \leftarrow h_{\ell(j)}(x_t^b) + W_j\left(s_j(x_t^s) - s_j(x_t^b)\right) = h_{\ell(j)}(x_t^b) + W_j W_j^\intercal\left(h_{\ell(j)}(x_t^s) - h_{\ell(j)}(x_t^b)\right).$$

Writing the output as $y_{j,t}^{\text{nn_swap}}$ and the factual output as $y_t^{\text{nn}} \coloneqq \mathfrak{N}(x_t^b)$, the *neural effect signature* is

$$\Delta_{j,t}^{\text{nn}} \coloneqq d^{\text{nn}}\big(y_{j,t}^{\text{nn_swap}}, y_t^{\text{nn}}\big), \qquad j = 1, \dots, n, \quad t = 1, \dots, T_{\text{ft}}.$$

We usually take $d^{\text{nn}}(y, y') = \text{softmax}\big(y\big) - \text{softmax}\big(y'\big)$, with other options including summaries based on the total variation distance $d^{\text{nn}}(y, y') = \frac{1}{2}\left\|\text{softmax}(y) - \text{softmax}(y')\right\|_1$, KL divergence, or any other summary of the neural base and swap outputs. See Appendix B.2 for an empirical comparison of several such effect signatures.

Step 2: Optimal transport matching. We associate each abstract variable and neural site with an effect signature, obtained by aggregating its output effects over the fit bank. We then fit an OT coupling between the signature collections, using UOT when irrelevant neural sites may be present. The coupling yields a weighted soft correspondence between abstract variables and neural sites.

Specifically, we pool effect signatures over fit pairs to obtain one signature per variable/site:

$$u_i \coloneqq \text{Agg}\big(\Delta_{i,1}^{\text{abs}}, \dots, \Delta_{i,T_{\text{ft}}}^{\text{abs}}\big), \qquad v_j \coloneqq \text{Agg}\big(\Delta_{j,1}^{\text{nn}}, \dots, \Delta_{j,T_{\text{ft}}}^{\text{nn}}\big).$$

For example, Agg can average or concatenate the per-pair effects, optionally followed by ℓ_2-normalization. These signatures define empirical measures $\mu \coloneqq \frac{1}{m}\sum_{i=1}^{m}\delta_{u_i}$ and $\nu \coloneqq \frac{1}{n}\sum_{j=1}^{n}\delta_{v_j}$, where δ_x is a Dirac mass. We compute the EOT coupling $\Pi_\varepsilon^\star \in \mathbb{R}_{\geq 0}^{m\times n}$ between μ and ν via Sinkhorn's algorithm with quadratic cost $c(u_i, v_j) = \|u_i - v_j\|^2$ (Cuturi, 2013). For UOT, we use the one-sided specialization described above, fixing $\pi_1 = \mu$ and relaxing only π_2, with parameter $\beta = \beta_2$. Slightly abusing notation, write Π for the fitted EOT or UOT coupling, whose i-th row $[\Pi]_{i:} \in \mathbb{R}^n$ gives the matching weights of Z_i to $s_1, \dots, s_n$.

Step 3: Handle calibration. Next, we use the fitted coupling Π and $\mathcal{D}_{\text{cal}}$ to calibrate an *intervention handle* for each abstract variable. A handle combines interventions at selected neural sites, with relative strengths determined by Π.[1] For abstract variable Z_i, calibration selects the number of neural sites K and a global intervention strength λ. Let $\mathcal{S}_i^K$ denote the indices of the K largest entries in the i-th row of Π. We restrict that row to $\mathcal{S}_i^K$ and renormalize $\widetilde{\Pi}_{j|i}^K \coloneqq \Pi_{ij}\mathbb{1}_{\{j\in\mathcal{S}_i^K\}}/\big(\sum_{j'\in\mathcal{S}_i^K}\Pi_{ij'}\big)$, for $j = 1, \dots, n$. Suppose first that the selected sites lie in one activation vector $h(x) \in \mathbb{R}^d$. Applied to a base/source pair $(x_t^b, x_t^s) \in \mathcal{D}_{\text{cal}}$, the corresponding handle performs

$$h \leftarrow h(x_t^b) + \lambda \sum_{j\in\mathcal{S}_i^K} \widetilde{\Pi}_{j|i}^K W_j\big(s_j(x_t^s) - s_j(x_t^b)\big).$$

When the selected sites lie in multiple layers, we apply the interventions in forward-pass order. At each later layer, the update is applied to the activation produced by the already-intervened computation.

Let $H_i^{K,\lambda}$ denote this candidate handle and let $y_{i,t}^{\text{nn_swap}}(K, \lambda)$ be the neural output obtained by applying it to calibration pair (x_t^b, x_t^s). We evaluate it using

$$\text{IIA}_{\text{cal}}(i; K, \lambda) \coloneqq \frac{1}{T_{\text{cal}}}\sum_{t=1}^{T_{\text{cal}}} \mathbb{1}_{\{y_{i,t}^{\text{nn_swap}}(K,\lambda)\sim y_{i,t}^{\text{abs_swap}}\}}, \tag{3}$$

where $\sim$ denotes task-dependent agreement between the neural and abstract intervention outputs. For an LLM, agreement holds when the highest-logit intervened token equals the abstract target. We select $(K_i^\star, \lambda_i^\star) \in \arg\max_{K\in\mathcal{K},\ \lambda\in\Lambda} \text{IIA}_{\text{cal}}(i; K, \lambda)$ and set the calibrated handle as $H_i^\star \coloneqq H_i^{K_i^\star,\lambda_i^\star}$. We repeat this for each abstract variable, equivalently, for each coupling row. EOT/UOT parameters and site resolutions, when searched, are also selected on $\mathcal{D}_{\text{cal}}$.

Step 4: Handle evaluation. Finally, we evaluate the IIA score, similarly to (3), for each calibrated intervention handle on the holdout test set $\mathcal{D}_{\text{te}}$, without further parameter selection. Unless stated otherwise, the IIA reported in our experiments is the average test IIA over all abstract variables.

Remark 3.1 (Alternative matching). OT provides a principled matching framework, which we adopt in PLOT. We note that other constructions of Π are possible, e.g., based on cosine similarities between effect signatures or direct IIA matching. We compare these alternatives with OT in Appendix B.1.

[1] In progressive localization, intermediate stages use calibration data to select parent sites, while the final stage calibrates the intervention handle.

3.2 Multi-Stage Progressive Localization

PLOT becomes hierarchical by composing Steps 1–2 across nested neural site families. Rather than matching all sites at the finest resolution at once, progressive localization first selects coarse parent sites, then fits new couplings within them. The number of stages depends on the architecture and desired granularity, as illustrated in Figure 1. The high-level causal model remains fixed throughout. The progressive stages refine neural resolution rather than define distinct levels of causal abstraction.

At stage r, let $\mathcal{T}_i^{(r)}$ denote the current candidate family for localizing Z_i. PLOT fits a coupling $\Pi_i^{(r)}$ between all target variables $Z_1, \ldots, Z_m$ and the sites in $\mathcal{T}_i^{(r)}$, using OT or one-sided UOT as in Step 2. The first stage uses a common candidate family and coupling for all variables. At each intermediate stage r, we take the highest-mass sites in the row $[\Pi_i^{(r)}]_{i:}$, evaluate their individual interventions on $\mathcal{D}_{\text{cal}}$, and select the site with the highest calibration IIA for further refinement. Subdividing that parent yields the finer candidate family $\mathcal{T}_i^{(r+1)}$ for the next stage. At the final stage r_{final}, Step 3 calibrates the i-th row of $\Pi_i^{(r_{\text{final}})}$ into an intervention handle for Z_i, and Step 4 evaluates it on $\mathcal{D}_{\text{te}}$. The final localization may also guide DAS, where PLOT selects coarse neural locations, such as a token position and layer, and then DAS learns the intervention subspace within that layer.

4 Experimental Settings and Results

Our experiments are organized by increasing localization complexity: (i) HEQ, where we test single-stage handle construction over individual neurons; (ii) binary addition, where we compare global and progressive localization of carry variables in native and PCA representations; and (iii) MCQA, where we examine accuracy under matched handle-dimension limits, runtime, and learned-handle structure, test UOT for broad candidate families with irrelevant sites, and compare against masking, PLOT-guided DAS, and full DAS. See Appendix A for full experiment details.

4.1 Hierarchical Equality (HEQ)

As a first single-stage test of PLOT, we consider the HEQ benchmark from Geiger et al. (2024), where the goal is to determine whether two pairs of objects have the same equality relation. On this small task, a single PLOT matching over all hidden neurons yields accurate, compact handles without progressive localization or learned intervention rotations.

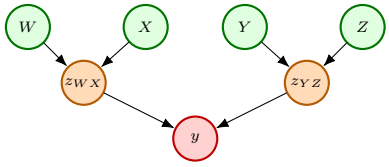


Figure 3: Causal model for HEQ.

Causal model. As in Figure 3, inputs $W, X, Y, Z \in [100]$ define equality variables $z_{WX} = \mathbb{1}_{\{W=X\}}$ and $z_{YZ} = \mathbb{1}_{\{Y=Z\}}$. The output $y = \mathfrak{C}(W, X, Y, Z) = \mathbb{1}_{\{z_{WX}=z_{YZ}\}}$ is 1 precisely when both pairs are equal or both are unequal. For example, $\mathfrak{C}(1, 1, 2, 2) = \mathfrak{C}(1, 2, 3, 4) = 1$, while $\mathfrak{C}(1, 1, 1, 2) = 0$.

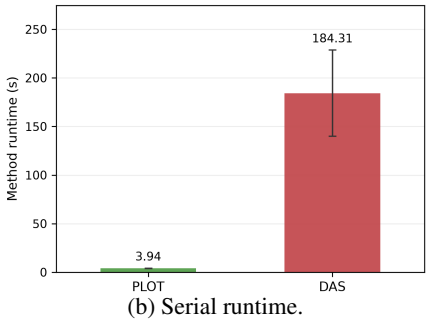
Neural model. We encode each of W, X, Y, Z with a fixed 4-dimensional embedding and concatenate them into an input $x \in \mathbb{R}^{16}$. The neural model is a ReLU MLP $\mathfrak{N} : \mathbb{R}^{16} \to \mathbb{R}^2$ with three hidden layers of width 16, trained to 100% accuracy on held-out factual data. See Appendix A.3 for details.

Single-stage PLOT. We use disjoint fit, calibration, and test pair banks; see Appendix A.3 for their construction. The abstract variables are $Z_1 = z_{WX}$ and $Z_2 = z_{YZ}$, while the candidate sites are all $3 \times 16 = 48$ individual hidden neurons. For the effect signatures, we encode abstract outputs as one-hot vectors and take $d^{\text{abs}}(y, y') = y - y'$, while for neural logits we take $d^{\text{nn}}(y, y') = \text{softmax}(y) - \text{softmax}(y')$, placing both signatures in the same $\mathbb{R}^2$ space. We fit the resulting 2×48 EOT coupling on $\mathcal{D}_{\text{ft}}$ using Sinkhorn's algorithm and calibrate the entropic parameter $\varepsilon \in \{0.01, 0.25, 2, 4, 8, 16\}$, support size $K \in \{1, \ldots, 16\}$, and intervention strength $\lambda \in \{1, \ldots, 16\}$ on $\mathcal{D}_{\text{cal}}$. As a baseline, DAS trains rotations on $\mathcal{D}_{\text{ft}}$ for each variable over layers $\ell = 1, 2, 3$ and subspace dimensions $r \in \{1, \ldots, 16\}$, and selects the best handle on $\mathcal{D}_{\text{cal}}$.

Results and learned handles. The table in Figure 4 reports test IIA over seeds 1–10 for z_{WX} and z_{YZ}, together with their average. For each abstract variable, we separately evaluate sensitivity, on pairs that change that variable, and invariance, on pairs that leave it fixed. Both single-stage PLOT and DAS recover accurate handles on this benchmark, with average test IIA 0.9931 and 0.9967, respectively. Thus, PLOT obtains high interchange intervention agreement directly over the original hidden neurons without learning a rotation. PLOT takes 3.94 seconds on average, compared with 184.31 seconds for the DAS layer and dimension search, a speedup of approximately $47\times$. Runtime

Metric	PLOT	DAS
z_{WX} sensitivity	0.9923 ± 0.0155	0.9938 ± 0.0041
z_{WX} invariance	0.9988 ± 0.0026	0.9998 ± 0.0004
z_{YZ} sensitivity	0.9913 ± 0.0190	0.9934 ± 0.0047
z_{YZ} invariance	0.9901 ± 0.0209	0.9999 ± 0.0003
Mean sensitive IIA	0.9918 ± 0.0172	0.9936 ± 0.0032
Avg. IIA	0.9931 ± 0.0127	0.9967 ± 0.0016
Serial runtime (s)	3.94 ± 0.27	184.31 ± 44.41

(a) Accuracy and serial runtime summary.



(b) Serial runtime.

Figure 4: HEQ comparison over 10 seeds. Values are mean ± standard deviation. Runtime excludes neural model training and includes fitting, calibration, and test evaluation.

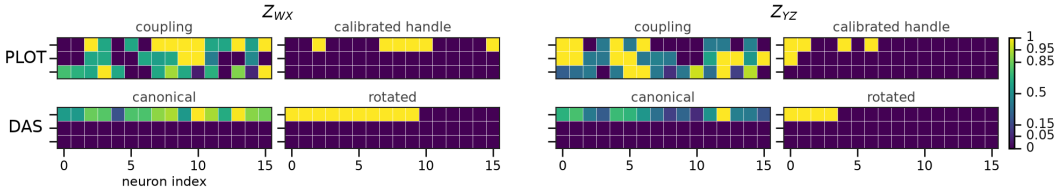


Figure 5: HEQ matching weights and intervention handles for one illustrative seed (seed 7). PLOT is shown before and after top-K calibration, and DAS in the canonical and rotated representations.

includes effect signature construction, coupling fitting or rotation training, calibration, and held-out evaluation, but excludes neural model training. See Appendix A.3 for hardware and timing details.

A structural difference between the methods is shown in Figure 5 for seed 7. DAS is fit layer by layer, whereas PLOT fits one transport plan over all candidate neurons simultaneously. Calibration converts the resulting matching weights into a compact top-K intervention handle. In contrast, DAS reaches similar performance through a learned rotated subspace. Further layerwise summaries, OT parameter sensitivity, and learned intervention sizes are reported in Appendix A.3.

4.2 Binary Addition

We next consider binary addition with a GRUCell-based recurrent network, where we seek to locate internal carry variables across timesteps and within recurrent states. We compare global and progressive localization in native and PCA coordinates, and show how PLOT can guide DAS without searching every timestep.

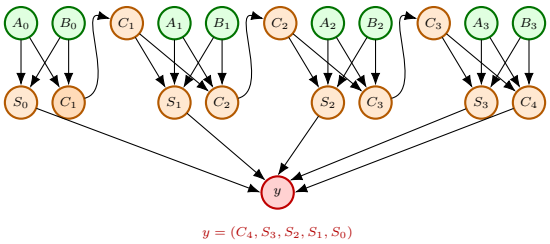


Figure 6: Causal model for 4-bit ripple-carry addition.

Causal model. As in Figure 6, the inputs are two 4-bit numbers $A_3A_2A_1A_0$ and $B_3B_2B_1B_0$. The abstract model is the standard ripple-carry adder, with $S_0 = (A_0 + B_0) \bmod 2$, $C_1 = \lfloor (A_0 + B_0)/2 \rfloor$, and for $i = 1, 2, 3$, $S_i = (A_i + B_i + C_i) \bmod 2$ and $C_{i+1} = \lfloor (A_i + B_i + C_i)/2 \rfloor$. The output is $y = (C_4, S_3, S_2, S_1, S_0)$. We report intervention accuracy for the internal carry variables C_1, C_2, C_3.

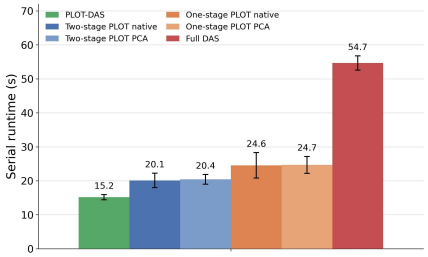
Neural model. The neural model reads bit-pairs (A_ℓ, B_ℓ) from least to most significant and updates a GRUCell state $h_\ell \in \mathbb{R}^{16}$ at timestep ℓ, for $\ell = 0, 1, 2, 3$. Linear readouts predict the five output bits $(C_4, S_3, S_2, S_1, S_0)$. The model is trained to solve 4-bit addition exactly; architecture and training details are in Appendix A.4. Thus, h_ℓ is the natural coarse neural site for carry $C_{\ell+1}$.

Global and progressive localization. We use the output-effect pipeline of Section 3.1 with binary-addition-specific pair banks and recurrent candidate sites. Following the MIB Arithmetic benchmark (Mueller et al., 2025), we construct structured base/source banks with operand flips and carry-targeting sources, split into fit, calibration, and test banks; see Appendix A.4. Candidate sites are native-coordinate blocks or PCA spans, with PCA fitted to fit-bank hidden activations at each timestep.

For each resolution, one-stage PLOT fits one coupling between the three carries and sites across all four timesteps, allowing handles to span timesteps. Two-stage PLOT first fits a 3×4 coupling over

Method	Test IIA	Serial runtime (s)
1-STAGE PLOT NATIVE	0.767 ± 0.032	24.6 ± 3.7
1-STAGE PLOT PCA	0.900 ± 0.053	24.7 ± 2.5
2-STAGE PLOT NATIVE	0.792 ± 0.023	20.1 ± 2.1
2-STAGE PLOT PCA	$\mathbf{0.936 \pm 0.028}$	20.4 ± 1.4
PLOT-DAS	$\mathbf{0.991 \pm 0.007}$	$\mathbf{15.2 \pm 0.8}$
Full DAS	$\mathbf{0.991 \pm 0.007}$	54.7 ± 2.1

(a) Accuracy and runtime summary.



(b) Serial runtime.

Figure 7: Binary-addition results over five seeds. Test IIA equally weights sensitivity and invariance and averages over C_1, C_2, C_3. Runtime covers the full parameter grids.

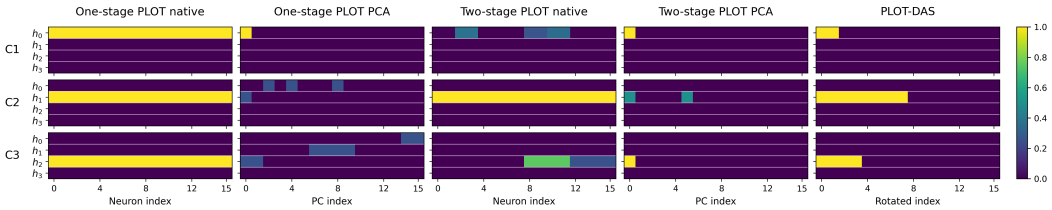


Figure 8: Binary-addition handles for one seed. Rows show carries and columns show methods.

the recurrent states to select one timestep per carry, then fits new couplings over finer sites within the selected states. Both use EOT, with ε, site resolution r, support size K, and strength λ selected on $\mathcal{D}_{\text{cal}}$. PLOT-guided DAS (PLOT-DAS) uses the coarse localization to train DAS only within selected states, while full DAS trains rotations at every timestep. Both select final handles on $\mathcal{D}_{\text{cal}}$.

Results and learned handles. Figure 7 compares test IIA and serial runtime over five seeds. Moving from native coordinates to PCA substantially improves direct PLOT under both search strategies, from 0.767 to 0.900 for one-stage PLOT and from 0.792 to 0.936 for two-stage PLOT. Two-stage localization also attains higher mean IIA than one-stage localization in both representations, while reducing runtime from about 24.6 seconds to 20.1–20.4 seconds. The strongest PLOT-only result is two-stage PLOT PCA, with test IIA 0.936 ± 0.028. PLOT-DAS reaches the same mean test IIA as full DAS, 0.991 ± 0.007, while reducing runtime from 54.7 to 15.2 seconds, a roughly $3.6\times$ speedup.

Figure 8 visualizes the learned handles for one illustrative run (seed 3). One-stage PLOT native selects the full 16-dimensional states h_0, h_1, h_2 for C_1, C_2, C_3, while one-stage PLOT PCA yields substantially smaller handles and can combine sites across timesteps. Two-stage PLOT first correctly localizes C_1, C_2, C_3 to h_0, h_1, h_2, respectively, and the PCA refinement yields compact handles of dimensions $1, 2, 1$, compared with $8, 16, 8$ in the native basis. PLOT-DAS selects the same timesteps but learns rotated subspaces of dimensions $2, 8, 4$. Full DAS is not shown, as it learns the exact same handles as PLOT-DAS through its exhaustive timestep sweep.

4.3 MULTIPLE-CHOICE QUESTION ANSWERING (MCQA)

We next evaluate PLOT in the substantially larger setting of the MCQA benchmark from MIB (Mueller et al., 2025), where Gemma-2-2B answers natural-language multiple-choice prompts. The model reads a color fact and query, identifies the answer containing the queried color, and outputs its symbol. For example, given *"The sky is blue. What color is the sky? (A) red (B) blue (C) green (D) yellow"*, the correct output is `B`. We compare one-stage and two-stage localization under a handle-dimension limit, together with masking, PLOT-DAS, and full DAS baselines.

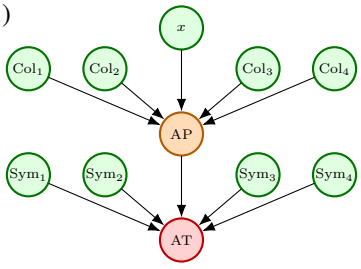


Figure 9: MCQA causal model, with abstract variables AP and AT.

Causal model. Each input is a natural-language prompt x containing a color fact, a query, and four answer choices. Let Col_i denote the color at choice position i, and Sym_i be the corresponding answer symbol, e.g., A, B, C, D. The abstract model computes the queried color $q(x)$, the answer pointer $\text{AP} = \arg\text{unique}_{i \in \{1,2,3,4\}}\{\text{Col}_i = q(x)\}$, and the answer token $\text{AT} = \text{Sym}_{\text{AP}}$; see Figure 9. The

Table 1: MCQA results for IIA and runtime over five seeds. Layer/dim columns report seed 0, where dimension denotes calibrated PLOT support, DBM feature count, or DAS subspace dimension.

Method	AP	AT	Avg.	Runtime (min)	AP layer/dim	AT layer/dim
1-STAGE PLOT NATIVE	0.4290 ± 0.3381	0.7620 ± 0.2958	0.5955 ± 0.2779	112.85 ± 2.64	L5/576	L20/1152
1-STAGE PLOT PCA	0.7640 ± 0.1130	0.9120 ± 0.0561	0.8380 ± 0.0435	124.71 ± 3.09	L14,17,20,21,23/705	L20,21,24/846
2-STAGE PLOT NATIVE	0.7630 ± 0.1026	0.9030 ± 0.0231	0.8330 ± 0.0523	49.11 ± 3.93	L17/512	L24/1024
2-STAGE PLOT PCA	0.7640 ± 0.1099	$\mathbf{0.9580 \pm 0.0115}$	0.8610 ± 0.0526	67.50 ± 4.34	L17/141	L24/282
DBM canonical	0.7880 ± 0.0582	0.6610 ± 0.0096	0.7245 ± 0.0264	13.63 ± 0.45	L17/1152	L23/1142
DBM PCA	0.7700 ± 0.0581	0.9440 ± 0.0129	0.8570 ± 0.0325	15.12 ± 0.36	L17/219	L23/246
DBM SAE	0.7130 ± 0.0512	0.7990 ± 0.0323	0.7560 ± 0.0142	14.96 ± 0.77	L20/109	L23/166
PLOT-DAS	0.8810 ± 0.0542	0.8690 ± 0.0500	0.8750 ± 0.0286	25.27 ± 1.60	L17/256	L24/128
Full DAS	$\mathbf{0.9220 \pm 0.0282}$	0.8700 ± 0.0296	$\mathbf{0.8960 \pm 0.0213}$	509.60 ± 23.88	L17/32	L20/512

abstract variables of interest are AP and AT, corresponding to *AnswerPointer* and *Answer* in MIB (Mueller et al., 2025). We evaluate counterfactuals that change AP, AT, or both; see Appendix A.5.

Neural model. We use Gemma-2-2B (Gemma Team, 2024), a decoder-only transformer with 26 residual-stream layers and width 2304. It receives the full MCQA prompt x and is evaluated by its next-token distribution over answer symbols. Localization is performed over final-token activations.

Global and progressive localization. We follow the pipeline of Section 4.2, replacing recurrent timesteps with final-token activations across 26 layers and using the MIB MCQA counterfactual families for fit, calibration, and test. One-stage and two-stage PLOT again use native-coordinate blocks or PCA spans. At the coarse stage, two abstract variables are matched to 26 layers, some of which may be irrelevant, so we use one-sided UOT to relax the neural marginal. Table 4 in the appendix shows consistently stronger UOT layer selection than OT, particularly for AP. Following MIB (Mueller et al., 2025), we limit each direct handle to 1152 dimensions, half the residual-stream width, summed across selected layers. We also compare PLOT-DAS, full DAS, and DBM in canonical, PCA, and SAE representations (Chaudhary & Geiger, 2024). See Appendix A.5 for more details.

Results and learned handles. Table 1 reports test IIA and serial runtime over five seeds. Two-stage PLOT improves mean IIA in both representations, from 0.5955 to 0.8330 in native coordinates and from 0.8380 to 0.8610 in PCA. For AP, native IIA rises from 0.429 to 0.763, while with PCA AP remains at 0.764 and AT increases from 0.912 to 0.958. Two-stage localization is also roughly $2\times$ faster than one-stage matching. The gap is larger than in binary addition due to the larger neural model and candidate family. PCA outperforms native coordinates under both strategies, with two-stage PLOT PCA giving the strongest direct PLOT result with the default signature, consistent with the binary-addition findings. An additional KL-signature ablation yields comparable or higher two-stage IIA with lower observed cross-seed variability but substantially weaker one-stage results; see Appendix B.2. DBM PCA is competitive at 0.8570, whereas the SAE representation considered in MIB is substantially weaker here at 0.7560. Full DAS remains the most accurate method, while PLOT-DAS retains most of this accuracy at 0.8750 and cuts runtime to 25.27 minutes, a roughly $20\times$ speedup. These MCQA results highlight PLOT's utility for accurate gradient-free handles and coarse localization that makes learned DAS substantially cheaper.

5 Discussion and Concluding Remarks

We proposed PLOT, a gradient-free approach for joint correspondence discovery via OT matching of intervention effects. The coupling can be calibrated into executable intervention handles or used progressively to restrict subsequent localization and guide subspace learning. Our experiments show that progressive localization can improve accuracy under handle-dimension limits and reduce computation, while PLOT-guided DAS approaches full-DAS accuracy at substantially lower cost.

Future directions include sharper localization rules that exploit finer resolutions and hierarchical signature constructions. Scaling PLOT to larger models will also require engineering around batching, activation caching, approximate signatures, and distributed transport computation (Wang & Goldfeld, 2026; Tsur et al., 2025). Unlike subspace methods such as DAS, PLOT's matching formulation need not operate directly on the neural network and can instead align a causal model with an interpretable feature representation thereof. A natural direction is to apply PLOT to features in Anthropic's cross-layer transcoder replacement models or even to nodes in their attribution graphs (Ameisen et al., 2025), with the goal of automating attribution graph discovery or compression.

6 AI USE STATEMENT

In this work, we used generative AI tools to assist with implementing, running, and debugging experiments; generating synthetic datasets; and reviewing the manuscript for typos, grammar, spelling, and notational consistency. We did not use generative AI tools to develop any research methodology or for writing any substantive prose. We have reviewed all AI-assisted work: for example, we checked that the algorithms implementing each of our paper's methods that were written by LLMs exactly match the pseudocode that we wrote ourselves and provided in the appendix, and ensured that reported results match saved experimental outputs. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Beatrice Acciaio, Julio Backhoff-Veraguas, and Anastasiia Zalashko. Causal optimal transport and its links to enlargement of filtrations and continuous-time stochastic optimization. *Stochastic Processes and their Applications*, 130(5):2918–2953, 2020. doi: 10.1016/j.spa.2019.08.009. URL https://doi.org/10.1016/j.spa.2019.08.009.

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

Sander Beckers and Joseph Y. Halpern. Abstracting causal models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 2678–2685, 2019. doi: 10.1609/aaai.v33i01.33012678. URL https://doi.org/10.1609/aaai.v33i01.33012678.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nicholas L. Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E. Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. URL https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Maheep Chaudhary and Atticus Geiger. Evaluating open-source sparse autoencoders on disentangling factual knowledge in GPT-2 Small. *CoRR*, abs/2409.04478, 2024. doi: 10.48550/arXiv.2409.04478. URL https://arxiv.org/abs/2409.04478.

Patrick Cheridito and Stephan Eckstein. Optimal transport and Wasserstein distances for causal models. *Bernoulli*, 31(2):1351–1376, 2025. doi: 10.3150/24-BEJ1773. URL https://doi.org/10.3150/24-BEJ1773.

Lénaïc Chizat, Gabriel Peyré, Bernhard Schmitzer, and François-Xavier Vialard. Scaling algorithms for unbalanced transport problems. *Mathematics of Computation*, 87(314):2563–2609, 2018.

Hoagy Cunningham, Aidan Ewart, Logan Riggs Smith, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. *CoRR*, abs/2309.08600, 2023. doi: 10.48550/arXiv.2309.08600. URL https://arxiv.org/abs/2309.08600.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*, volume 26, 2013.

Yorgos Felekis, Fabio Massimo Zennaro, Nicola Branchini, and Theodoros Damoulas. Causal optimal transport of abstractions. In *Proceedings of the Third Conference on Causal Learning and Reasoning*, volume 236 of *Proceedings of Machine Learning Research*, pp. 462–498, 2024. URL https://proceedings.mlr.press/v236/felekis24a.html.

Wilfrid Gangbo and Andrzej Święch. Optimal maps for the multidimensional Monge–Kantorovich problem. *Communications on Pure and Applied Mathematics*, 51(1):23–45, 1998.

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

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 *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 7324–7338, 2022. URL https://proceedings.mlr.press/v162/geiger22a.html.

Atticus Geiger, Zhengxuan Wu, Christopher Potts, Thomas Icard, and Noah D. Goodman. Finding alignments between interpretable causal variables and distributed neural representations. In *Proceedings of the Third Conference on Causal Learning and Reasoning*, volume 236 of *Proceedings of Machine Learning Research*, pp. 160–187, 2024. URL https://proceedings.mlr.press/v236/geiger24a.html.

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

Gemma Team. Gemma 2: Improving open language models at a practical size. *CoRR*, abs/2408.00118, 2024. doi: 10.48550/arXiv.2408.00118. URL https://arxiv.org/abs/2408.00118.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=F76bwRSLeK.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. URL https://mlanthology.org/iclr/2015/kingma2015iclr-adam/.

Rémi Lassalle. Causal transport plans and their Monge–Kantorovich problems. *Stochastic Analysis and Applications*, 36(3):452–484, 2018. doi: 10.1080/07362994.2017.1422747. URL https://doi.org/10.1080/07362994.2017.1422747.

Samuel Marks and Max Tegmark. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=aajyHYjjsk.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, volume 35, 2022. doi: 10.52202/068431-1262. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/6f1d43d5a82a37e89b0665b33bf3a182-Abstract-Conference.html.

Aaron Mueller, Atticus Geiger, Sarah Wiegreffe, Dana Arad, Iván Arcuschin, Adam Belfki, Yik Siu Chan, Jaden Fried 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 *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 45069–45108. PMLR, 2025. URL https://proceedings.mlr.press/v267/mueller25a.html.

Judea Pearl. *Causality*. Cambridge University Press, 2nd edition, 2009.

Gabriel Peyré and Marco Cuturi. *Computational optimal transport*. Now Publishers, 2019.

Gregory Polyakov, Christian Hepting, Carsten Eickhoff, and Seyed Ali Bahrainian. Interpretability analysis of arithmetic in-context learning in large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 1758–1777, 2025. doi: 10.18653/v1/2025.emnlp-main.92. URL https://aclanthology.org/2025.emnlp-main.92/.

Paul K. Rubenstein, Sebastian Weichwald, Stephan Bongers, Joris M. Mooij, Dominik Janzing, Moritz Grosse-Wentrup, and Bernhard Schölkopf. Causal consistency of structural equation models. In *Proceedings of the 33rd Conference on Uncertainty in Artificial Intelligence (UAI)*, 2017. URL http://auai.org/uai2017/proceedings/papers/11.pdf.

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. URL https://openreview.net/forum?id=6fDjUoEQvm.

Curt Tigges, Oskar John Hollinsworth, Atticus Geiger, and Neel Nanda. Linear representations of sentiment in large language models. *CoRR*, abs/2310.15154, 2023. doi: 10.48550/arXiv.2310.15154. URL https://arxiv.org/abs/2310.15154.

Dor Tsur, Ziv Goldfeld, Kristjan Greenewald, and Haim Permuter. Neural estimation for scaling entropic multimarginal optimal transport, 2025. URL https://arxiv.org/abs/2506.00573.

Cédric Villani. *Topics in optimal transportation*, volume 58 of *Graduate Studies in Mathematics*. American Mathematical Society, 2003.

Tao Wang and Ziv Goldfeld. Neural entropic optimal transport and Gromov–Wasserstein alignment. *IEEE Transactions on Information Theory*, 72(4):2424–2443, 2026. doi: 10.1109/TIT.2026.3661439. URL https://doi.org/10.1109/TIT.2026.3661439.

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*, volume 36, 2023. doi: 10.52202/075280-3418. URL https://papers.neurips.cc/paper_files/paper/2023/hash/f6a8b109d4d4fd64c75e94aaf85d9697-Abstract-Conference.html.

Appendix

A Experimental Details

A.1 Evaluation Conventions

After calibrating K and λ for each abstract variable Z_i, we evaluate on separate holdout test sets $\mathcal{D}_{\text{te}}$:

- $\boldsymbol{Z_i}$**-sensitivity set:** containing pairs (x^b, x^s) for which x^s changes the value of Z_i relative to x^b.
- $\boldsymbol{Z_i}$**-invariance set:** containing pairs (x^b, x^s) for which x^s leaves the value of Z_i fixed.

We evaluate each calibrated handle separately on both sets, checking both sensitivity to interventions that change Z_i and invariance to interventions that leave Z_i unchanged. This differs from prior works (Geiger et al., 2024; Wu et al., 2023; Mueller et al., 2025) which mixed sensitive and invariant pairs, leading to test accuracies that are less interpretable. Indeed, a method that learns no handle can score perfectly on invariance-heavy test sets, yielding high average accuracy while being weak on sensitivity. Unless stated otherwise, main-text accuracies average over all $2m$ sensitive and invariant test sets (two for each variable $Z_1, \ldots, Z_m$).

A.2 Runtime Conventions

We report serial method runtime per seed, starting after loading or training the factual model and obtaining data banks. Runtime includes method-specific preparation (such as effect signatures and PCA), coupling computation or DAS training, calibration over the reported search grid, and evaluation of the selected handle on the held-out test bank. For certain multi-stage methods, we may reuse handles and runtimes obtained from shared initial stages. Reported method runtime therefore underestimates the full wall-clock time of the experiment.

A.3 Hierarchical Equality (HEQ)

Neural backbone training. We encode each tuple $(W, X, Y, Z) \in [100]^4$ using a fixed embedding map $e : [100] \to \mathbb{R}^4$ and concatenate the embeddings into $x = [e(W)\ e(X)\ e(Y)\ e(Z)] \in \mathbb{R}^{16}$. The neural model is a ReLU MLP with three hidden layers of width 16, resulting in 850 trainable parameters. We train the MLP with cross-entropy loss using Adam (Kingma & Ba, 2015), learning rate 10^{-3}, and batch size 1024 for three epochs on 1,048,576 factual examples, matching the setup in Geiger et al. (2024). The model reaches exact accuracy 1.0 on a disjoint validation set of size 10,000.

Pair bank construction. We construct the HEQ fit, calibration, and test banks as follows:

- **Fit bank:** 1000 pairs from a mixed policy with roughly 50% pairs sensitive in either z_{WX} or z_{YZ}, and 50% invariant in both.
- **Calibration bank:** 1000 pairs, half only z_{WX}-sensitive and half only z_{YZ}-sensitive.
- **Test banks:** four holdout banks of size 1000, one sensitive and one invariant for each of z_{WX} and z_{YZ}.

Effect signature construction. We let $d^{\text{abs}}(y, y') = y - y'$ where outputs are one-hot over two classes, and $d^{\text{nn}}(y, y') = \text{softmax}(y) - \text{softmax}(y')$ where outputs are logits over two classes. Thus we compute $\Delta^{\text{abs}}_{i,t}, \Delta^{\text{nn}}_{j,t} \in \mathbb{R}^2$ for each of the $T_{\text{ft}} = 1000$ fit pairs. Next, we take Agg as simply concatenation of the effects in a common fit-pair order without averaging or normalization, giving $u_i, v_j \in \mathbb{R}^{2000}$.

Hardware and runtime details. We run all methods locally on an Apple M1 Pro with 8 CPU cores and 16GB of memory. Runtime excludes factual-model training and includes the full parameter searches:

- **PLOT:** runtime includes effect signature construction over $\mathcal{D}_{\text{ft}}$ for the two abstract variables and $3 \times 16 = 48$ neural sites, Sinkhorn fitting of an EOT coupling for each $\varepsilon \in \{0.01, 0.25, 2, 4, 8, 16\}$, calibration of each coupling row over $K \in \{1, \ldots, 16\}$ and $\lambda \in \{1, \ldots, 16\}$ on $\mathcal{D}_{\text{cal}}$, and final evaluation on $\mathcal{D}_{\text{te}}$ of the calibrated intervention. This gives $2 \times 6 \times 16 \times 16 = 3072$ calibration settings per seed across the two variables.

Table 2: HEQ results for seed 7, used to generate Figure 5. Serial runtime includes the full searches, calibration, and test evaluation, but excludes factual model training and model/bank loading.

Method	z_{WX}	z_{YZ}	Average	Serial runtime (s)	Best setting
PLOT	0.9980	0.9930	0.9955	3.80	$\varepsilon = 2$, $K = (6, 5)$, $\lambda = (6, 5)$
DAS	0.9960	0.9900	0.9930	184.04	Layer-1 sites with subspace dimensions 10 for z_{WX} and 4 for z_{YZ}

- **DAS:** for learning rate 10^{-3} and maximum epoch budget $N = 1000$ with early stopping, runtime includes training rotations on $\mathcal{D}_{\text{ft}}$ for each abstract variable, hidden layer $\ell \in \{1, 2, 3\}$, and subspace dimension $r \in \{1, \ldots, 16\}$, calibration over all (ℓ, r) pairs on $\mathcal{D}_{\text{cal}}$, and final evaluation of the selected rotation on $\mathcal{D}_{\text{te}}$.

The accuracy and runtime in Figure 4 report means and standard deviations across seeds 1–10.

Heatmap figure details. To generate Figure 5, we select one HEQ seed (namely, seed 7) to display learned handles. The accuracies and serial runtimes for this seed are displayed in Table 2.

The PLOT panels show matching weights over the 3×16 hidden neurons of the HEQ MLP. For PLOT before calibration, we show the coupling rows for z_{WX} and z_{YZ} over the 48 candidate sites, renormalized to $[0, 1]$ for visualization. For PLOT after calibration, we display only the weights on the calibrated top-K support, again renormalized to $[0, 1]$. Calibration selects the shared entropic parameter $\varepsilon = 2$, with $(K, \lambda) = (6, 6)$ for z_{WX} and $(5, 5)$ for z_{YZ}.

For DAS, we display the best calibration-selected rotation after sweeping layers ℓ and subspace dimensions r. For z_{WX}, $\ell = 1$ and $r = 10$, while for z_{YZ}, $\ell = 1$ and $r = 4$. In the canonical basis panel, we visualize each neuron's contribution to the selected intervention subspace, normalized to $[0, 1]$. In the rotated basis panel, we mark the first r rotated coordinates in layer ℓ, which are the actual coordinates intervened on by DAS.

Effect of entropic regularization. We test sensitivity to the entropic regularization parameter ε in (1) by rerunning the HEQ PLOT pipeline on the same backbone and calibration protocol while varying only ε. At each fixed ε, we select K and λ on $\mathcal{D}_{\text{cal}}$ and evaluate the resulting handles on the held-out test banks. As shown in Figure 10, smaller values of ε perform best, with mean sensitive-test IIA remaining near 0.996 for $\varepsilon = 2^j$, $j = -9, \ldots, -5$. Performance decreases as the regularization grows, with the largest drop around $\varepsilon = 2^3$, before partially recovering for larger values.

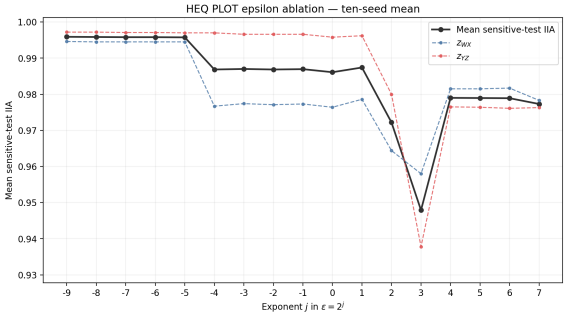


Figure 10: Sensitivity to the entropic parameter ε, averaged over ten seeds.

A.4 Binary Addition

Neural backbone training. We use a single-layer GRUCell recurrent network, which utilizes one gated recurrent unit (GRU) cell[2]. The input consists of two 4-bit numbers

$$a = (a_0, a_1, a_2, a_3), \qquad b = (b_0, b_1, b_2, b_3),$$

processed from least to most significant bit. At timestep ℓ,

$$x_\ell = [a_\ell, b_\ell] \in \mathbb{R}^2, \qquad h_\ell = \text{GRUCell}(x_\ell, h_{\ell-1}) \in \mathbb{R}^d, \qquad h_{-1} = 0.$$

[2] https://docs.pytorch.org/docs/2.11/generated/torch.nn.GRUCell.html

The hidden state therefore carries information from lower-order bits to higher-order bits, enabling carry propagation. Sum-bit logits are read from each timestep,

$$y_\ell^{(S)} = w_S^\intercal h_\ell + b_S, \qquad \ell = 0, 1, 2, 3,$$

and the final carry logit is read from the last state,

$$y^{(C)} = w_C^\intercal h_3 + b_C, \qquad y = [y^{(C)}, y_3^{(S)}, y_2^{(S)}, y_1^{(S)}, y_0^{(S)}] \in \mathbb{R}^5.$$

We train a model with $d = 16$ using binary cross entropy and Adam at learning rate 2×10^{-2}, batch size 64, and 250 epochs with early stopping. The model achieves exact accuracy on all $2^4 \times 2^4 = 256$ addition pairs.

Pair bank construction. There are $2^4 \times 2^4 = 256$ possible inputs, which we collect as a bank of base examples. For each base example, we use a policy to sample 26 source examples:

- 8 source examples where each one flips the value of one of the 8 input bits, $a_0, \ldots, a_3, b_0, \ldots, b_3$.
- 18 carry-targeted source examples: 3 that change the value of C_1, 5 for C_2, 7 for C_3, and 3 for C_4.

We use a $128 - 64 - 64$ split for the 256 base examples, which means $128 \times 26 = 3328$ pairs for the fit set, $64 \times 26 = 1664$ for the calibration set, and $64 \times 26 = 1664$ for the test set.

For the test set, we record for each pair whether or not it changed the value of C_1, C_2, and C_3, the three abstract variables of interest. Then, we partition the 1664 test pairs into sensitive and invariant sets for each of these variables.

Effect signature construction. We let $d^{\text{abs}}(y, y') = y - y'$ where outputs are five-bit binary numbers, and $d^{\text{nn}}(y, y') = \sigma(y) - \sigma(y')$ where outputs are logits for each of the five bits. Thus we compute $\Delta_{i,t}^{\text{abs}}, \Delta_{j,t}^{\text{nn}} \in \mathbb{R}^5$ for each of the $T_{\text{ft}} = 3328$ fit pairs. The fit bank contains 26 source-construction families with 128 pairs per family, so we take Agg to be the mean five-dimensional effect within each family, concatenate over the 26 means, and ℓ_2-normalize the resulting vector. Thus $u_i, v_j \in \mathbb{R}^{130}$ for full-timestep, native-block, and PCA-band sites.

Hardware, runtime, and algorithm details. We run all methods locally on an Apple M1 Pro with 8 CPU cores and 16GB of memory. The algorithm and runtime for each method are detailed below.

Algorithm 1 Binary Addition: 1-STAGE PLOT NATIVE

Require: Splits $\mathcal{D}_{\text{ft}}$, $\mathcal{D}_{\text{cal}}$, and $\mathcal{D}_{\text{te}}$, factual pre-trained GRUCell with hidden size d

```
Start timer
Construct effect signatures over D_ft for abstract carry variables C_1, C_2, C_3
for resolution r ∈ {1, 2, 4, ..., d} do
    Partition h_0, ..., h_3 into neural sites with resolution r
    Construct effect signatures over D_ft for the partitioned neural sites
    for ε ∈ {0.003, 0.01, 0.03, 0.1} do
        Compute the Sinkhorn EOT coupling in R^{3×4⌊d/r⌋} using ε
        for carry variable C_i ∈ {C_1, C_2, C_3} do
            for (top-K, λ) ∈ {1, 2, 4} × {0.25, 0.5, 1, 2, 4, 8} do
                Evaluate the candidate intervention handle H_i^{K,λ} for C_i on D_cal
            end for
        end for
    end for
end for
Evaluate the best calibrated handles H_1*, ..., H_3* on D_te
End timer and report 1-STAGE PLOT NATIVE runtime
```

Algorithm 2 Binary Addition: 1-STAGE PLOT PCA

Require: Splits $\mathcal{D}_{\text{ft}}$, $\mathcal{D}_{\text{cal}}$, and $\mathcal{D}_{\text{te}}$, factual pre-trained GRUCell with hidden size d

Start timer
Construct effect signatures over $\mathcal{D}_{\text{ft}}$ for abstract carry variables C_1, C_2, C_3
Collect hidden states from $\mathcal{D}_{\text{ft}}$ for each timestep $h_0, \ldots, h_3$
Compute the centered PCA rotations with full rank; store them as $R_0, \ldots, R_3$
for band count $b \in \{1, 2, 4, \ldots, d\}$ **do**
    Partition the d PCA directions for each timestep into b disjoint neural sites
    Set W_j to be the columns of R_{h_j} corresponding to s_j, where h_j is the timestep containing s_j
    Construct effect signatures over $\mathcal{D}_{\text{ft}}$ for the PCA-band neural sites
    for $\varepsilon \in \{0.003, 0.01, 0.03, 0.1\}$ **do**
        Compute the Sinkhorn EOT coupling in $\mathbb{R}^{3\times 4b}$ using ε
        for carry variable $C_i \in \{C_1, C_2, C_3\}$ **do**
            for (top-K, λ) $\in \{1, 2, 4\} \times \{0.25, 0.5, 1, 2, 4, 8\}$ **do**
                Evaluate the candidate intervention handle $H_i^{K,\lambda}$ for C_i on $\mathcal{D}_{\text{cal}}$
            end for
        end for
    end for
end for
Evaluate the best calibrated handles $H_1^\star, \ldots, H_3^\star$ on $\mathcal{D}_{\text{te}}$
End timer and report 1-STAGE PLOT PCA runtime

Algorithm 3 Binary Addition: 2-STAGE PLOT NATIVE

Require: Splits $\mathcal{D}_{\text{ft}}$, $\mathcal{D}_{\text{cal}}$, and $\mathcal{D}_{\text{te}}$, factual pre-trained GRUCell with hidden size d

Start timer
Stage 1:
Construct effect signatures over $\mathcal{D}_{\text{ft}}$ for the four neural timestep activations $h_0, \ldots, h_3$
Construct effect signatures over $\mathcal{D}_{\text{ft}}$ for abstract carry variables C_1, C_2, C_3
for $\varepsilon \in \{0.01, 0.03\}$ **do**
    Compute the Sinkhorn EOT coupling in $\mathbb{R}^{3\times 4}$ using ε
    for carry variable $C_i \in \{C_1, C_2, C_3\}$ **do**
        Extract the top timestep ℓ_i for row C_i
        Evaluate timestep h_{ℓ_i} for C_i on $\mathcal{D}_{\text{cal}}$
    end for
end for
Choose $\{\ell_1, \ell_2, \ell_3\}$ that achieves highest mean calibration score across choices of ε
Stage 2:
for $\ell \in \{\ell_1, \ell_2, \ell_3\}$ **do**
    for resolution $r \in \{1, 2, 4, \ldots, d\}$ **do**
        Partition h_ℓ into neural sites with resolution r
        Construct effect signatures over $\mathcal{D}_{\text{ft}}$ for the partitioned neural sites
        for $\varepsilon \in \{0.003, 0.01, 0.03, 0.1\}$ **do**
            Compute the Sinkhorn EOT coupling in $\mathbb{R}^{3\times\lfloor d/r \rfloor}$ using ε
            for C_i with top timestep ℓ **do**
                for (top-K, λ) $\in \{1, 2, 4\} \times \{0.25, 0.5, 1, 2, 4, 8\}$ **do**
                    Evaluate the candidate intervention handle $H_i^{K,\lambda}$ for C_i on $\mathcal{D}_{\text{cal}}$
                end for
            end for
        end for
    end for
end for
Evaluate the best calibrated handles $H_1^\star, \ldots, H_3^\star$ on $\mathcal{D}_{\text{te}}$
End timer and report 2-STAGE PLOT NATIVE runtime

Algorithm 4 Binary Addition 2-STAGE PLOT PCA

Require: Splits $\mathcal{D}_{\text{ft}}$, $\mathcal{D}_{\text{cal}}$, and $\mathcal{D}_{\text{te}}$, factual pre-trained GRUCell with hidden size d

Start timer
Stage 1:
Construct effect signatures over $\mathcal{D}_{\text{ft}}$ for the four neural timestep activations $h_0, \ldots, h_3$
Construct effect signatures over $\mathcal{D}_{\text{ft}}$ for abstract carry variables C_1, C_2, C_3
for $\varepsilon \in \{0.01, 0.03\}$ **do**
    Compute the Sinkhorn EOT coupling in $\mathbb{R}^{3\times 4}$ using ε
    for carry variable $C_i \in \{C_1, C_2, C_3\}$ **do**
        Extract the top timestep ℓ_i for row C_i
        Evaluate timestep h_{ℓ_i} for C_i on $\mathcal{D}_{\text{cal}}$
    end for
end for
Choose $\{\ell_1, \ell_2, \ell_3\}$ that achieves the highest mean calibration score across choices of ε
Stage 2:
for $\ell \in \{\ell_1, \ell_2, \ell_3\}$ **do**
    Collect hidden states from $\mathcal{D}_{\text{ft}}$ at timestep h_ℓ
    Compute the centered PCA rotation with full rank; store it as R_ℓ
    for band count $b \in \{1, 2, 4, \ldots, d\}$ **do**
        Partition the d PCA directions at h_ℓ into b disjoint neural sites
        For each j, set W_j to be the columns of R_ℓ corresponding to the PCA-band s_j
        Construct effect signatures over $\mathcal{D}_{\text{ft}}$ for the PCA-band neural sites
        for $\varepsilon \in \{0.003, 0.01, 0.03, 0.1\}$ **do**
            Compute the Sinkhorn EOT coupling in $\mathbb{R}^{3\times b}$ using ε
            for C_i with top timestep ℓ **do**
                for (top-K, λ) $\in \{1, 2, 4\} \times \{0.25, 0.5, 1, 2, 4, 8\}$ **do**
                    Evaluate the candidate intervention handle $H_i^{K,\lambda}$ for C_i on $\mathcal{D}_{\text{cal}}$
                end for
            end for
        end for
    end for
end for
Evaluate the best calibrated handles $H_1^\star, \ldots, H_3^\star$ on $\mathcal{D}_{\text{te}}$
End timer and report 2-STAGE PLOT PCA runtime

Algorithm 5 Binary Addition PLOT-DAS

Require: DAS learning rate $\mathrm{lr} = 10^{-2}$, DAS maximum epochs $N = 100$
Require: Splits $\mathcal{D}_{\mathrm{ft}}$, $\mathcal{D}_{\mathrm{cal}}$, and $\mathcal{D}_{\mathrm{te}}$, factual pre-trained GRUCell with hidden size d
Start timer
Stage 1:
Construct effect signatures over $\mathcal{D}_{\mathrm{ft}}$ for the four neural timestep activations $h_0, \ldots, h_3$
Construct effect signatures over $\mathcal{D}_{\mathrm{ft}}$ for abstract carry variables C_1, C_2, C_3
for $\varepsilon \in \{0.01, 0.03\}$ **do**
    Compute the Sinkhorn EOT coupling in $\mathbb{R}^{3\times 4}$ using ε
    for carry variable $C_i \in \{C_1, C_2, C_3\}$ **do**
        Extract the top timestep ℓ_i for row C_i
        Evaluate timestep h_{ℓ_i} for C_i on $\mathcal{D}_{\mathrm{cal}}$
    end for
end for
Choose $\{\ell_1, \ell_2, \ell_3\}$ that achieves the highest mean calibration score across choices of ε
Stage 2:
for $C_i \in \{C_1, C_2, C_3\}$ **do**
    for $k \in \{1, 2, 4, \ldots, d\}$ **do**
        Train a DAS rotation for C_i on $\mathcal{D}_{\mathrm{ft}}$ at timestep h_{ℓ_i} with subspace intervention size k
        Evaluate the rotation handle for C_i on $\mathcal{D}_{\mathrm{cal}}$
    end for
end for
Evaluate the best calibrated rotations for C_1, C_2, C_3 on $\mathcal{D}_{\mathrm{te}}$
End timer and report PLOT-DAS runtime

Algorithm 6 Binary Addition Full DAS

Require: DAS learning rate $\mathrm{lr} = 10^{-2}$, DAS maximum epochs $N = 100$
Require: Splits $\mathcal{D}_{\mathrm{ft}}$, $\mathcal{D}_{\mathrm{cal}}$, and $\mathcal{D}_{\mathrm{te}}$, factual pre-trained GRUCell with hidden size d
Start timer
for $C_i \in \{C_1, C_2, C_3\}$ **do**
    for timestep $\ell \in \{0, 1, 2, 3\}$ **do**
        for $k \in \{1, 2, 4, \ldots, d\}$ **do**
            Train a DAS rotation for C_i on $\mathcal{D}_{\mathrm{ft}}$ at timestep h_ℓ with subspace intervention size k
            Evaluate the rotation handle for C_i on $\mathcal{D}_{\mathrm{cal}}$
        end for
    end for
end for
Evaluate the best calibrated rotations for C_1, C_2, C_3 on $\mathcal{D}_{\mathrm{te}}$
End timer and report Full DAS runtime

A.5 MULTIPLE-CHOICE QUESTION ANSWERING (MCQA)

Pair bank construction. For the MCQA task, MIB-Bench (Mueller et al., 2025) open-sources a dataset[3] used to train and test their methods. However, this public dataset only contains 210 rows, so we synthetically generate a dataset[4] with 10,000 rows in the same format as their dataset. See Table 3 for a base example and associated counterfactuals. Recall that the causal model in Figure 9 contains two abstract variables: AP and AT. As in MIB-Bench, we consider the following three counterfactuals:

- `answerPosition`: permute the positions of the answer choice colors $\mathrm{Col_i}$, without changing the answer choice symbols $\mathrm{Sym_i}$. This typically affects both AP and AT.
- `randomLetter`: randomly sample four different letters of the alphabet to replace the answer choice symbols $\mathrm{Sym_i}$. This typically affects AT.
- `answerPosition + randomLetter`: apply both counterfactuals above at the same time. This typically affects both AP and AT.

Before constructing pair banks, we filter examples to ensure that Gemma-2-2B correctly completes the prompts before any interventions. We first sample 2,000 rows from our dataset, forming $2{,}000 \times 3 = 6{,}000$ (base, source) pairs from the three counterfactuals. Then we pass each pair into Gemma-2-2B: a (base, source) pair passes the filter if Gemma-2-2B correctly completes both the base and source examples, up to capitalization, whitespace, and punctuation. Finally, we sample the following fit, calibration, and test banks:

- Fit bank: 200 pairs sampled randomly from the filtered bank.
- Calibration banks: 200 sensitive pairs for each AP and AT.
- Test banks: 200 sensitive pairs for each AP and AT.

We ensure that no two banks contain (base, source) pairs with the same base question.

Table 3: An MCQA example and selected associated counterfactuals.

Counterfactual	**Text**	**Correct Completion**
Original Prompt	*Question: A banana is yellow. What color is a banana? A. red\nB. blue\nC. yellow\nD. green\nAnswer:*	C
`answerPosition`	*Question: A banana is yellow. What color is a banana? A. green\nB. yellow\nC. red\nD. blue\nAnswer:*	B
`randomLetter`	*Question: A banana is yellow. What color is a banana? H. red\nT. blue\nP. yellow\nX. green\nAnswer:*	P
`answerPosition + randomLetter`	*Question: A banana is yellow. What color is a banana? H. green\nT. yellow\nP. red\nX. blue\nAnswer:*	T

Effect signature construction. See Appendix B.2, where we compare three variants of signature construction as an ablation experiment—family-wise, KL, and TV. The experiments in our main text utilize the family-wise signature.

OT vs. UOT details. In Table 4, we compare OT and UOT as Stage-A layer-selection methods for the final-token residual stream. Each method produces a 2×26 coupling, with rows corresponding to AP and AT and columns corresponding to transformer layers. For each row, we take the top-6 layers by mass, calibrate the associated full-layer interventions on that abstract variable, and report the layer

[3] `https://huggingface.co/datasets/mib-bench/copycolors_mcqa`

[4] The dataset will be made publicly available upon publication; it is omitted here for anonymity.

with the highest calibration accuracy. Because standard OT must distribute mass across all 26 layer sites using only two abstract rows, it yields weak layer selection for AP. In contrast, UOT can leave irrelevant layers unmatched, which allows it to select high-scoring AP layers across all five seeds.

Table 4: Selected MCQA layers and accuracies for Stage A OT vs UOT across five seeds.

	OT		UOT	
Seed	AP	AT	AP	AT
0	L6 (0.000)	L24 (0.970)	L17 (0.765)	L24 (0.970)
1	L6 (0.000)	L24 (0.975)	L17 (0.775)	L24 (0.975)
2	L6 (0.000)	L24 (0.965)	L17 (0.770)	L24 (0.965)
3	L6 (0.000)	L24 (0.975)	L19 (0.695)	L24 (0.975)
4	L6 (0.000)	L24 (0.985)	L19 (0.685)	L24 (0.985)

Intervention agreement relation. For $Z \in \{\mathrm{AP}, \mathrm{AT}\}$, the abstract intervention output $y_{i,t}^{\mathrm{abs_swap}}$ is a single uppercase answer letter for pair (x_t^b, x_t^s). After applying a handle H_i to the base prompt, let $L_{H_i,t}(v)$ denote the next-token logit assigned to token $v \in \mathcal{V}$. For MCQA, the task-dependent agreement relation $\sim$ in (3) is

$$y_{H_i}^{\mathrm{nn_swap}}(x_t^b, x_t^s) \sim y_{i,t}^{\mathrm{abs_swap}} \quad \Longleftrightarrow \quad \mathcal{N}\left(\arg\max_{v \in \mathcal{V}} L_{H_i,t}(v)\right) = y_{i,t}^{\mathrm{abs_swap}},$$

where $\mathcal{V}$ is the full next-token vocabulary. The function $\mathcal{N}$ decodes the token then applies Unicode NFKC normalization, trims surrounding whitespace, and folds case to uppercase; it returns an invalid symbol unless the result is exactly one letter in A–Z. We use this agreement relation to compute the fraction of correct pairs in each target's 200-pair test bank. The reported overall MCQA IIA is the arithmetic mean of the AP and AT test accuracies.

We note that this differs from MIB's evaluation procedure, which instead checks whether the entire decoded continuation at all contains the expected answer letter preceded by a space. This substring criterion does not require the continuation to be exactly the answer letter: it can count a longer output containing that letter as correct. We find this protocol too relaxed, and therefore report held-out IIA using exact agreement after normalizing the highest-logit token to a single letter.

Hardware, runtime, and algorithm details. We run all seeds and methods on NVIDIA A40 GPUs. The algorithm and runtime for each method are detailed below.

Algorithm 7 MCQA: 1-STAGE PLOT NATIVE

Require: Splits $\mathcal{D}_{\mathrm{ft}}$, $\mathcal{D}_{\mathrm{cal}}$, and $\mathcal{D}_{\mathrm{te}}$, factual pre-trained Gemma-2-2B
Start timer
Construct effect signatures over $\mathcal{D}_{\mathrm{ft}}$ for abstract variables AP, AT
for resolution $r \in \{32, 64, 128, 256, 576, 1152\}$ **do**
    Partition final-token residual layers $\ell_0, \ldots, \ell_{25}$ into neural sites with resolution r
    Construct effect signatures over $\mathcal{D}_{\mathrm{ft}}$ for the partitioned neural sites
    for $(\varepsilon, \beta_{\mathrm{neural}}) \in \{0.5\} \times \{0.1, 0.3, 1, 3\}$ **do**
        Compute the Sinkhorn UOT coupling in $\mathbb{R}^{2\times 26\lfloor 2304/r \rfloor}$ using ε and β_{neural}
        for $Z \in \{\mathrm{AP}, \mathrm{AT}\}$ **do**
            for (top-K, λ) $\in \{1, 2, 3, 4, 5\} \times \{0.5, 1, 2, 4, 8, 12, 20, 30\}$ **do**
                if handle size does not exceed 1152 neurons **then**
                    Evaluate the candidate intervention handle $H^{K,\lambda}$ for Z on $\mathcal{D}_{\mathrm{cal}}$
                end if
            end for
        end for
    end for
end for
Evaluate the best calibrated handles $H^\star_{\mathrm{AP}}, H^\star_{\mathrm{AT}}$ on $\mathcal{D}_{\mathrm{te}}$
End timer and report 1-STAGE PLOT NATIVE runtime

Algorithm 8 MCQA: 1-STAGE PLOT PCA

Require: Splits $\mathcal{D}_{\text{ft}}$, $\mathcal{D}_{\text{cal}}$, and $\mathcal{D}_{\text{te}}$, factual Gemma-2-2B
Start timer
Construct effect signatures over $\mathcal{D}_{\text{ft}}$ for abstract variables AP, AT
Collect hidden states from $\mathcal{D}_{\text{ft}}$ for each final-token residual layer $\ell_0, \ldots, \ell_{25}$
Compute the centered PCA rotations with retained empirical rank; store them as $R_0, \ldots, R_{25}$
for band count $b \in \{1, 2, 4, 8, 16, 32, 64\}$ **do**
    Partition each layer's retained PCA directions into b disjoint neural sites
    Set W_j to be the columns of R_{ℓ_j} corresponding to s_j, where ℓ_j is the layer containing s_j
    Construct effect signatures over $\mathcal{D}_{\text{ft}}$ for the PCA-band neural sites
    for $(\varepsilon, \beta_{\text{neural}}) \in \{0.5\} \times \{0.1, 0.3, 1, 3\}$ **do**
        Compute the Sinkhorn UOT coupling in $\mathbb{R}^{2\times 26b}$ using ε and β_{neural}
        for $Z \in \{\text{AP}, \text{AT}\}$ **do**
            for (top-K, λ) $\in \{1, 2, 3, 4, 5\} \times \{0.5, 1, 2, 4, 8, 12, 20, 30\}$ **do**
                if handle size does not exceed 1152 PCA directions **then**
                    Evaluate the candidate intervention handle $H^{K,\lambda}$ for Z on $\mathcal{D}_{\text{cal}}$
                end if
            end for
        end for
    end for
end for
Evaluate the best calibrated handles $H^{\star}_{\text{AP}}, H^{\star}_{\text{AT}}$ on $\mathcal{D}_{\text{te}}$
End timer and report 1-STAGE PLOT PCA runtime

Algorithm 9 MCQA: 2-STAGE PLOT NATIVE

Require: Splits $\mathcal{D}_{\text{ft}}$, $\mathcal{D}_{\text{cal}}$, and $\mathcal{D}_{\text{te}}$, factual pre-trained Gemma-2-2B
Start timer
Stage 1:
Construct effect signatures over $\mathcal{D}_{\text{ft}}$ for final-token residual layers $\ell_0, \ldots, \ell_{25}$
Construct effect signatures over $\mathcal{D}_{\text{ft}}$ for abstract variables AP, AT
for $(\varepsilon, \beta_{\text{neural}}) \in \{0.5\} \times \{0.1, 0.3, 1, 3\}$ **do**
    Compute the Sinkhorn UOT coupling in $\mathbb{R}^{2\times 26}$ using ε and β_{neural}
    for $Z \in \{\text{AP}, \text{AT}\}$ **do**
        Extract the six highest-mass layers for row Z
        Evaluate their full-layer interventions for Z on $\mathcal{D}_{\text{cal}}$; retain highest-scoring layer ℓ_Z
    end for
end for
Choose $\{\ell_{\text{AP}}, \ell_{\text{AT}}\}$ that achieves highest mean calibration score across choices of ε and β_{neural}
Stage 2:
for $\ell \in \{\ell_{\text{AP}}, \ell_{\text{AT}}\}$ **do**
    for resolution $r \in \{32, 64, 128, 256, 576, 1152\}$ **do**
        Partition ℓ into neural sites with resolution r
        Construct effect signatures over $\mathcal{D}_{\text{ft}}$ for the partitioned neural sites
        for $(\varepsilon, \beta_{\text{neural}}) \in \{0.5\} \times \{0.1, 0.3, 1, 3\}$ **do**
            Compute the Sinkhorn UOT coupling in $\mathbb{R}^{2\times\lfloor 2304/r \rfloor}$ using ε and β_{neural}
            for $Z \in \{\text{AP}, \text{AT}\}$ with top layer ℓ **do**
                for (top-K, λ) $\in \{1, 2, 3, 4, 5\} \times \{0.5, 1, 2, 4, 8, 12, 20, 30\}$ **do**
                    if handle size does not exceed 1152 neurons **then**
                        Evaluate the candidate intervention handle $H_Z^{K,\lambda}$ on $\mathcal{D}_{\text{cal}}$
                    end if
                end for
            end for
        end for
    end for
end for
Evaluate the best calibrated handles $H^{\star}_{\text{AP}}, H^{\star}_{\text{AT}}$ on $\mathcal{D}_{\text{te}}$
End timer and report 2-STAGE PLOT NATIVE runtime

Algorithm 10 MCQA: 2-STAGE PLOT PCA

Require: Splits $\mathcal{D}_{\mathrm{ft}}$, $\mathcal{D}_{\mathrm{cal}}$, and $\mathcal{D}_{\mathrm{te}}$, factual pre-trained Gemma-2-2B
Start timer
Stage 1:
Construct effect signatures over $\mathcal{D}_{\mathrm{ft}}$ for final-token residual layers $\ell_0, \ldots, \ell_{25}$
Construct effect signatures over $\mathcal{D}_{\mathrm{ft}}$ for abstract variables AP, AT
for $(\varepsilon, \beta_{\mathrm{neural}}) \in \{0.5\} \times \{0.1, 0.3, 1, 3\}$ **do**
    Compute the Sinkhorn UOT coupling in $\mathbb{R}^{2\times 26}$ using ε and β_{neural}
    for $Z \in \{\mathrm{AP}, \mathrm{AT}\}$ **do**
        Extract the six highest-mass layers for row Z
        Evaluate their full-layer interventions for Z on $\mathcal{D}_{\mathrm{cal}}$; retain highest-scoring layer ℓ_Z
    end for
end for
Choose $\{\ell_{\mathrm{AP}}, \ell_{\mathrm{AT}}\}$ that achieves highest mean calibration score across choices of ε and β_{neural}
Stage 2:
for $\ell \in \{\ell_{\mathrm{AP}}, \ell_{\mathrm{AT}}\}$ **do**
    Collect hidden states from $\mathcal{D}_{\mathrm{ft}}$ for final-token residual layer ℓ
    Compute the centered PCA basis with retained empirical rank; store it as R_ℓ
    for band count $b \in \{1, 2, 4, 8, 16, 32, 64\}$ **do**
        Partition the retained PCA directions at ℓ into b disjoint neural sites
        Set W_j to be the columns of R_ℓ corresponding to s_j, for each PCA-band s_j
        Construct effect signatures over $\mathcal{D}_{\mathrm{ft}}$ for the PCA-band neural sites
        for $(\varepsilon, \beta_{\mathrm{neural}}) \in \{0.5\} \times \{0.1, 0.3, 1, 3\}$ **do**
            Compute the Sinkhorn UOT coupling in $\mathbb{R}^{2\times b}$ using ε and β_{neural}
            for $Z \in \{\mathrm{AP}, \mathrm{AT}\}$ with top layer ℓ **do**
                for (top-K, λ) $\in \{1, 2, 3, 4, 5\} \times \{0.5, 1, 2, 4, 8, 12, 20, 30\}$ **do**
                    if handle size does not exceed 1152 PCA directions **then**
                        Evaluate the candidate intervention handle $H_Z^{K,\lambda}$ on $\mathcal{D}_{\mathrm{cal}}$
                    end if
                end for
            end for
        end for
    end for
end for
Evaluate the best calibrated handles $H^\star_{\mathrm{AP}}, H^\star_{\mathrm{AT}}$ on $\mathcal{D}_{\mathrm{te}}$
End timer and report 2-STAGE PLOT PCA runtime

Algorithm 11 MCQA PLOT-DAS

Require: DAS learning rate $\mathrm{lr} = 10^{-3}$, DAS maximum epochs $N = 100$
Require: Splits $\mathcal{D}_{\mathrm{ft}}$, $\mathcal{D}_{\mathrm{cal}}$, and $\mathcal{D}_{\mathrm{te}}$, factual pre-trained Gemma-2-2B
Start timer
Stage 1:
Construct effect signatures over $\mathcal{D}_{\mathrm{ft}}$ for final-token residual layers $\ell_0, \ldots, \ell_{25}$
Construct effect signatures over $\mathcal{D}_{\mathrm{ft}}$ for abstract variables AP, AT
for $(\varepsilon, \beta_{\mathrm{neural}}) \in \{0.5, 1, 2, 4\} \times \{0.1, 0.3, 1, 3\}$ **do**
    Compute the Sinkhorn UOT coupling in $\mathbb{R}^{2\times 26}$ using ε and β_{neural}
    for $Z \in \{\mathrm{AP}, \mathrm{AT}\}$ **do**
        Extract the six highest-mass layers for row Z
        Evaluate their full-layer interventions for Z on $\mathcal{D}_{\mathrm{cal}}$; retain highest-scoring layer ℓ_Z
    end for
end for
Choose $\{\ell_{\mathrm{AP}}, \ell_{\mathrm{AT}}\}$ that achieves highest mean calibration score across choices of ε and β_{neural}
Stage 2:
for $Z \in \{\mathrm{AP}, \mathrm{AT}\}$ **do**
    for $k \in \{16, 32, 64, 96, 128, 256, 512, 768, 1024, 1152\}$ **do**
        Train a DAS rotation for Z on $\mathcal{D}_{\mathrm{ft}}$ at layer ℓ_Z with subspace intervention size k
        Evaluate the rotation handle for Z on $\mathcal{D}_{\mathrm{cal}}$
    end for
end for
Evaluate the best calibrated rotations for AP and AT on $\mathcal{D}_{\mathrm{te}}$
End timer and report PLOT-DAS runtime

Algorithm 12 MCQA Full DAS

Require: DAS learning rate $\mathrm{lr} = 10^{-3}$, DAS maximum epochs $N = 100$
Require: Splits $\mathcal{D}_{\mathrm{ft}}$, $\mathcal{D}_{\mathrm{cal}}$, and $\mathcal{D}_{\mathrm{te}}$, factual pre-trained Gemma-2-2B
Start timer
for $Z \in \{\mathrm{AP}, \mathrm{AT}\}$ **do**
    for $\ell \in \{\ell_0, \ldots, \ell_{25}\}$ **do**
        for $k \in \{16, 32, 64, 96, 128, 256, 512, 768, 1024, 1152\}$ **do**
            Train a DAS rotation on $\mathcal{D}_{\mathrm{ft}}$ at layer ℓ with subspace intervention size k
            Evaluate the rotation handle for Z on $\mathcal{D}_{\mathrm{cal}}$
        end for
    end for
end for
Evaluate the best calibrated rotation for AP and AT on $\mathcal{D}_{\mathrm{te}}$
End timer and report Full DAS runtime

B Additional Ablations

B.1 Non-OT Matching Alternatives

Although we use OT as the matching method of choice throughout the paper, this section compares it with two non-OT alternatives for defining the matching weights Π:

- **Cosine matching:** Collect effect signatures as before, then define pairwise scores using their cosine similarity:
$$A_{ij}^{\cos} := \frac{u_i^\intercal v_j}{\|u_i\|_2 \|v_j\|_2},$$
for nonzero signatures u_i and v_j.
- **IIA matching:** For each abstract variable i and neural site j, directly compute their alignment quality using interchange intervention accuracy on $\mathcal{D}_{\text{ft}}$:
$$A_{ij}^{\text{IIA}} := \frac{1}{T_{\text{ft}}} \sum_{t=1}^{T_{\text{ft}}} \mathbb{1}_{\{y_{j,t}^{\text{nn_swap}} \sim y_{i,t}^{\text{abs_swap}}\}}.$$

For either score matrix A, we apply a row-wise softmax to obtain the matching weights used for localization and handle calibration:

$$\Pi_{ij} = \frac{\exp(A_{ij})}{\sum_{j'=1}^{n} \exp(A_{ij'})}.$$

We compare UOT, cosine, and IIA matching for 2-STAGE PLOT NATIVE on MCQA, applying each method during both Stage A layer localization and Stage B within-layer native handle extraction. For UOT and cosine, we evaluate both the default family-wise signature used in the main text and the KL signature, whose construction is given in Appendix B.2 below. IIA matching instead uses fit-bank intervention accuracy directly and does not require effect signatures. Because IIA is also our evaluation metric, this construction is intentionally evaluation-aligned, while test IIA remains held out. We therefore view it as a strong non-OT comparator.

Table 5 displays the results over five seeds. KL-based UOT achieves the highest mean overall IIA at 0.8620. UOT also outperforms cosine under both signatures, yielding 0.8330 versus 0.8000 with the family-wise signature and 0.8620 versus 0.8305 with KL. IIA matching reaches mean IIA 0.8530, between family-wise UOT and KL-based UOT. Cosine matching is faster than UOT under both signatures and avoids its transport-parameter sweeps (namely, ε and β), while IIA matching lies between them in runtime. Thus, non-OT alternatives also produce accurate handles within the same progressive, gradient-free pipeline, with different accuracy–runtime tradeoffs. We adopt OT as the default because it provides a global constrained matching formulation, accommodates marginal relaxation through UOT, and draws on well-developed statistical and computational foundations, while the ablation shows competitive empirical performance.

Table 5: 2-STAGE PLOT NATIVE MCQA matching ablation over five seeds. Values are mean $\pm$ standard deviation. Runtime is serial time; layer/dim columns report the selected seed-0 handles.

Method	AP	AT	Avg.	Runtime (min)	AP layer/dim	AT layer/dim
UOT (family-wise)	0.7630 ± 0.1026	0.9030 ± 0.0231	0.8330 ± 0.0523	49.11 ± 3.93	L17/512	L24/1024
Cosine (family-wise)	0.6960 ± 0.0820	0.9040 ± 0.0258	0.8000 ± 0.0359	$\mathbf{15.60 \pm 0.42}$	L19/768	L24/1152
UOT (KL)	$\mathbf{0.8250 \pm 0.0298}$	0.8990 ± 0.0216	$\mathbf{0.8620 \pm 0.0174}$	40.85 ± 1.92	L18/512	L24/1152
Cosine (KL)	0.7580 ± 0.0635	0.9030 ± 0.0275	0.8305 ± 0.0349	16.50 ± 0.60	L18/1152	L24/1152
IIA matching	0.7850 ± 0.0491	$\mathbf{0.9210 \pm 0.0279}$	0.8530 ± 0.0161	23.79 ± 0.72	L18/768	L24/1024

B.2 Effect Signature Ablation

As an ablation study, we compare three effect signatures on MCQA: the family-wise signature on a truncated alphabet vocabulary used in the main text, a total-variation (TV) signature on the full output vocabulary, and a Kullback–Leibler (KL) signature on the full output vocabulary.

- **Family-wise signature (default):** First, we define the abstract effect on fit pair t as

$$\Delta_{i,t}^{\text{abs}} = y_{i,t}^{\text{abs_swap}} - y_t^{\text{abs}} \in \mathbb{R}^{26},$$

where each abstract output is a one-hot encoding over the uppercase letters A–Z. We define the effect of neural site s_j using raw logits:

$$\Delta_{j,t}^{\text{nn}} = y_{j,t}^{\text{nn_swap}} - y_t^{\text{nn}} \in \mathbb{R}^{26},$$

where outputs are Gemma-2-2B's logits restricted to the tokens of the uppercase letters A–Z. The Agg operation is defined as follows. Let I_f contain the fit pairs from $\mathcal{D}_{\text{ft}}$ in counterfactual family f, defined in Appendix A.5. For both abstract and neural effects, we average within each family, center across the 26 coordinates, and ℓ_2-normalize:

$$\bar{\Delta}_f = \frac{1}{|I_f|} \sum_{t \in I_f} \Delta_t, \qquad \widetilde{\Delta}_f = \bar{\Delta}_f - \frac{\mathbf{1}^\top \bar{\Delta}_f}{26} \mathbf{1}, \qquad z_f = \begin{cases} \widetilde{\Delta}_f / \|\widetilde{\Delta}_f\|_2, & \|\widetilde{\Delta}_f\|_2 > 0, \\ \widetilde{\Delta}_f, & \text{otherwise.} \end{cases}$$

Finally, we concatenate the three family blocks, giving

$$u_i^{\text{fam}} = \left[z_{i,\text{answerPosition}}^{\text{abs}}, z_{i,\text{randomLetter}}^{\text{abs}}, z_{i,\text{answerPosition_randomLetter}}^{\text{abs}} \right] \in \mathbb{R}^{78},$$

and

$$v_j^{\text{fam}} = \left[z_{j,\text{answerPosition}}^{\text{nn}}, z_{j,\text{randomLetter}}^{\text{nn}}, z_{j,\text{answerPosition_randomLetter}}^{\text{nn}} \right] \in \mathbb{R}^{78}.$$

- **TV signature:** We define the abstract effect signature to record whether the abstract intervention changes the output:

$$u_i^{\text{TV}} = \left[\mathbb{1}_{\left\{ y_{i,t}^{\text{abs_swap}} \neq y_t^{\text{abs}} \right\}} \right]_{t=1}^{T_{\text{ft}}} \in \mathbb{R}^{200},$$

where $T_{\text{ft}} = 200$. The intervention effect of neural site s_j on fit pair t is

$$\Delta_{j,t}^{\text{TV}} = \frac{1}{2} \left\| \text{softmax}\left(y_{j,t}^{\text{nn_swap}} \right) - \text{softmax}(y_t^{\text{nn}}) \right\|_1,$$

where the neural logits and softmax distributions now span the full output vocabulary. Letting Agg concatenate these effects in fit-pair order gives the neural signature

$$v_j^{\text{TV}} = \left[\Delta_{j,1}^{\text{TV}}, \Delta_{j,2}^{\text{TV}}, \ldots, \Delta_{j,T_{\text{ft}}}^{\text{TV}} \right] \in \mathbb{R}^{200}.$$

- **KL signature:** The abstract signature is the same output-change indicator vector,

$$u_i^{\text{KL}} = u_i^{\text{TV}} \in \mathbb{R}^{200}.$$

We define the effect of neural site s_j on fit pair t as

$$\Delta_{j,t}^{\text{KL}} = 1 - \exp\left(-\mathsf{KL}\left(\text{softmax}\left(y_{j,t}^{\text{nn_swap}} \right) \,\middle\|\, \text{softmax}(y_t^{\text{nn}}) \right) \right),$$

with softmax distributions over the full output vocabulary. We use the transformation $1 - \exp(-D_{\text{KL}})$ to bound the raw KL value to $[0, 1)$, then concatenate via Agg to obtain

$$v_j^{\text{KL}} = \left[\Delta_{j,1}^{\text{KL}}, \Delta_{j,2}^{\text{KL}}, \ldots, \Delta_{j,T_{\text{ft}}}^{\text{KL}} \right] \in \mathbb{R}^{200}.$$

These variants differ in their output vocabulary, effect statistic, and aggregation, so the ablation compares complete signature constructions.

We compare the performance of these signatures under one-stage and two-stage PLOT in native and PCA representations, with TV reported only for two-stage native PLOT. For two-stage methods, each signature is used both for coarse layer localization and within-layer handle extraction. Table 6 displays the results over five seeds.

With two-stage native PLOT, KL attains the highest mean test IIA at 0.8620, compared with 0.8440 for TV and 0.8330 for the family-wise signature. With two-stage PCA, KL gives a smaller increase, from 0.8610 to 0.8655. In both representations, KL yields lower observed cross-seed variability in overall IIA and lower mean runtime than the family-wise signature.

Without staging, however, KL reduces mean IIA from 0.5955 to 0.0085 in native coordinates and from 0.8380 to 0.4840 in PCA. Thus, all three tested signatures yield accurate handles with two-stage PLOT, but signature choice interacts strongly with localization strategy.

Table 6: MCQA effect-signature ablation with UOT over five seeds. Values are mean ± standard deviation. Family-wise is the default signature used in the main text; TV is reported only for two-stage native PLOT. Bold marks the highest mean overall IIA within each method group.

Method	Signature	AP	AT	Avg.	Runtime (min)
1-STAGE PLOT NATIVE	Family-wise	0.4290 ± 0.3381	0.7620 ± 0.2958	$\mathbf{0.5955 \pm 0.2779}$	112.85 ± 2.64
	KL	0.0100 ± 0.0035	0.0070 ± 0.0057	0.0085 ± 0.0034	131.06 ± 4.07
1-STAGE PLOT PCA	Family-wise	0.7640 ± 0.1130	0.9120 ± 0.0561	$\mathbf{0.8380 \pm 0.0435}$	124.71 ± 3.09
	KL	0.5970 ± 0.2690	0.3710 ± 0.1104	0.4840 ± 0.1559	140.87 ± 4.03
2-STAGE PLOT NATIVE	Family-wise	0.7630 ± 0.1026	0.9030 ± 0.0231	0.8330 ± 0.0523	49.11 ± 3.93
	TV	0.7820 ± 0.0609	0.9060 ± 0.0282	0.8440 ± 0.0423	41.04 ± 1.60
	KL	0.8250 ± 0.0298	0.8990 ± 0.0216	$\mathbf{0.8620 \pm 0.0174}$	40.85 ± 1.92
2-STAGE PLOT PCA	Family-wise	0.7640 ± 0.1099	0.9580 ± 0.0115	0.8610 ± 0.0526	67.50 ± 4.34
	KL	0.7710 ± 0.0323	0.9600 ± 0.0117	$\mathbf{0.8655 \pm 0.0152}$	61.73 ± 1.63