Same Data, Different Target: How the Prediction Target Can Separate Probing from Steering

Anonymous authors
Paper under double-blind review

Abstract

A linear probe can decode a fact while steering along it barely changes the model's output, and such a failure is easily read as the model not using the fact. We ask whether it can instead reflect what the model's readout, its output layer, was trained to predict. We train small blocks-world transformers that share data, initialisation and batch order and differ only in whether the readout predicts the current, next or previous state. Under the next- or previous-state target, clear(x) stays decodable, but its difference-in-means direction aligns more with the readout row of holding(x) than with its own. In our random play, holding(x) is possible one step away only if clear(x) holds now, and a model fitted to the labels alone predicts where the direction goes. How much steering along it changes the clear(x) output depends on its component on that output's row relative to the outputs' logit margins: it is partial where next-state outputs are uncertain and, in distribution at our primary strength, near zero where previous-state outputs are confident, under 3% of what the row itself achieves. The alignment replicates on new seeds and on one encoding whose action counts do not fix the state, and in Othello the target sets the sign of a probe direction's alignment with its own row. A steering failure can therefore reflect the readout's target rather than the representation; where the readout is hidden, a probe fitted to the model's own outputs diagnoses the alignment.

1 Introduction

Interpretability research often reads a concept with a linear probe and then tests it by steering along that probe. Under the linear representation hypothesis a concept is a direction (Park et al., 2024), so we might expect the direction that reads a fact to also control it. Yet probe accuracy can reflect the probe rather than the representation (Hewitt & Liang, 2019), and decodable information need not be used (Elazar et al., 2021). Accurate decoding also need not find a direction that controls the output (Billa, 2026; Galeone et al., 2026), a decodability–steerability gap. Existing work measures or repairs this gap in given models, by interpolating toward the control direction (Galeone et al., 2026), repairing readout rows (Garcia, 2026) or choosing layers (Wang et al., 2026). In all of this work, what the readout was trained to predict stays fixed.

Suppose a probe reads clear(B) almost perfectly, yet pushing the activations along it barely changes the output named clear(B). We would naturally conclude that the model does not use the fact. But if that output was trained on a shifted target, the probe direction may have turned toward the readout row of a neighbouring fact, such as holding(B). The failure would then tell us about the readout's target, and little about the representation. We therefore vary only the target, because it decides what a steering failure means.

To do so, we train state-tracking transformers that are identical in data, initialisation and batch order, and whose affine readouts differ only in *when* they predict the state: now, one action later or one action earlier (Section 2). We find that shifting the target *rotates* the difference-in-means (DiM) probe direction of clear(x), in the sense that its cosines with the readout rows change between the matched models (Figure 1): the direction moves toward the holding(x) row. The two facts are coupled across one step in both directions, since picking x up requires clear(x) and putting x down leaves it clear.

Our contributions are as follows.

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.

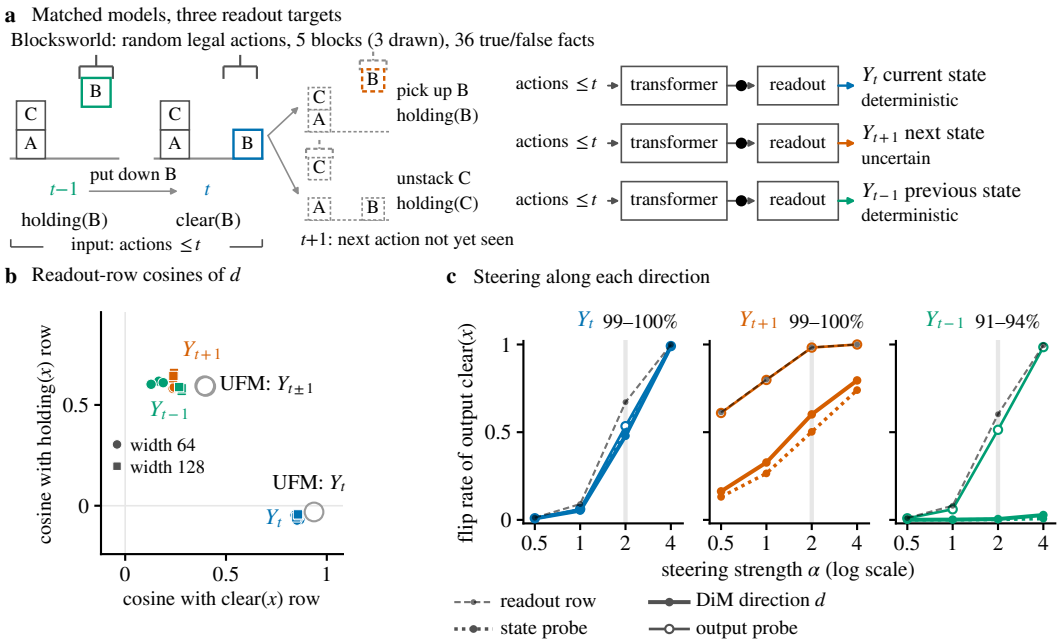

Figure 1: **Changing only the readout's target moves d toward the holding(x) row; steering along d stays partial under Y_{t+1} and nearly vanishes under Y_{t-1}. (a)** Matched models differ only in the readout's target; probes are fitted at the dot. **(b)** Cosines of d, the DiM direction for clear(x), with two readout rows, one mark per model; rings: UFM predictions. **(c)** Flip rate, in distribution, of the output clear(x), which reports clear$_{t\pm1}(x)$ under $Y_{t\pm1}$, at width 64 (width 128: Table 2). Lines: means over 3 seeds; percentages: state-probe accuracy; vertical line: $\alpha = 2$.

- Under the next- and previous-state targets, clear(x)'s DiM direction aligns more with the holding(x) row than with its own, and clear(x) stays decodable. The rotation replicates on new seeds and on one encoding that removes a counting shortcut of the task (Sections 3 and 5.1).
- An unconstrained features model (UFM), fitted to the labels alone, predicts where the direction goes (Section 3).
- The direction's projection on the readout row and the outputs' margins set how much it steers: partly where next-state outputs are uncertain, and near zero where previous-state outputs are confident. Probes fitted to the model's own outputs diagnose this alignment, and steering one block earlier shows that later computation reads the direction, including through the coupled output (Section 4).
- In Othello, the target sets the sign of the occupancy DiM direction's alignment with its readout row (Section 5).

2 Setup

Task and models. The models track the state of a fully observable blocks world. At every position of an action sequence they output 36 binary predicates, such as clear(x) and holding(x) for each block x. Each predicate is a deterministic function of the action prefix, so we know every probe's target exactly. The task also admits a *counting shortcut*. The state equals the initial state plus the action counts times fixed per-action effects, so it can be read from action counts alone (Appendix F).

Our primary models are depth-2 transformers of width $64/128$ and context 60, in 6 matched sets of three (Section 3). We also use a sweep of 27 state-tracking models over several depths and widths in Sections 4.1 and 4.3, and Othello models in Section 5. Appendix A gives every recipe.

A worked episode. We follow one step of the episode drawn in Figure 1a. At step $t-1$ the hand holds block B. The action "put down B" places B on the table, so at step t the predicate clear(B) is true. Each readout must give a different answer at position t. A current-state readout reports this state, with clear(B) true. A previous-state readout reports the state before the action, in which B is still held and therefore not clear. A next-state readout must report the state after an action it has not yet seen. If that action picks up B, the answer includes holding(B). If it unstacks C from A, the answer includes holding(C) instead, and B stays clear on the table.

The next- and previous-state targets tie clear(B) to holding(B). A block can be picked up only if it is clear, so clear(B) now is evidence about holding(B) one step later. Our episodes are *random legal play*, with each action drawn uniformly from the legal ones (Appendix A). Every such action fills or empties the hand, so a block held one step earlier has just been put down or stacked and is clear now. In this play, clear(x) at step t is therefore necessary for holding(x) at steps $t+1$ and $t-1$, and under either target the output named holding(x) reports a fact that clear(x) now informs. The coupling lies in the labels of this play, so a model of the labels alone can test it (Section 3.2).

Affine readout. The *readout* is the model's trained affine output layer. Let h be the residual stream at the probed site, and let W_j be the *readout row* of output bit j, its last-layer weight vector. The row is the per-bit analogue of an unembedding vector (Park et al., 2024). Then

$$z_j(h) = W_j^\top h + b_j, \qquad z_j(h+\delta) - z_j(h) = W_j^\top \delta \tag{1}$$

exactly, for any δ. The direction the readout uses for bit j is therefore a row of trained weights, and we can compute an intervention's effect on that bit as one inner product with it. Equation 1 is exact wherever δ enters directly before an affine readout, as in the state-tracking and Othello models.

Matched models. We compare *matched models* in sets of three that share initialisation, training episodes, minibatch order, optimiser, schedule and update count. Their readouts are trained on the state at the current position, Y_t (the *current-state target*), one action later, Y_{t+1} (*next-state*), or one action earlier, Y_{t-1} (*previous-state*). Any two of them form a *matched pair*, and we call the next- and previous-state targets *shifted*. The next state is uncertain, since the next action is not yet seen, while the current and previous states are deterministic. Because everything upstream of the loss is shared, we can attribute a within-pair difference to the target.

Probe directions. We compare three directions at the probed site. The *difference-in-means (DiM) direction* $d_j = \mathbb{E}[h \mid Y_{t,j}] - \mathbb{E}[h \mid \neg Y_{t,j}]$, unit-normalised, is the mass-mean probe of Marks & Tegmark (2024). The *state probe* w_{state} is a logistic probe on the true current state. Both d and w_{state} follow the true state. The *output probe* w_{output} is instead a logistic probe fitted to the model's own output bits, so it recovers the direction the readout uses, like the output-aligned concepts of Billa (2026). We measure geometry by a direction's cosine with a readout row, and we say that the target *rotates* a direction when these cosines change between matched models. Only the cosines are frame-free: whether the direction or the clear(x) row turns depends on how the two models' coordinates are aligned; we describe the rotation in the shared-initialisation frame, in which the direction turns (a post hoc comparison, Appendix D).

Steering and margins. We steer by activation addition (Turner et al., 2023; Panickssery et al., 2024). A unit direction is added as δ with norm αs, where s is the activation scale and α the *steering strength*. We then count how often an answer changes. In the state-tracking and Othello models we steer after the last block, directly before the readout, so this steering tests alignment with the readout; Section 4.4 steers one block earlier to test whether later computation uses the direction. The *flip rate* ρ is the fraction of all evaluated bits that take the targeted value, opposite the model's own output. It is a logit flip in the sense of Tigges et al. (2024), and in Othello it is the rate of argmax changes. Under the next- or previous-state target the output named clear(x) reports clear$_{t\pm1}(x)$, and we score steering on that output. By equation 1 the readout row flips the most at a given norm, so we also report a direction's flip rate as a fraction of the row's, ρ/ρ_{row}.

Why can a direction that decodes a fact fail to steer it? Decoding depends on how well the direction separates the two classes, but steering depends on something else. By equation 1, adding δ changes the logit of output j by $W_j^\top \delta$, so only the component of δ along row j matters. That component flips the output only if it carries the logit across zero, a distance set by the margin $|z_j|$. For a unit direction

with cosine c on row j, added with norm αs, the output therefore flips exactly when

$$\alpha c > u_j, \qquad u_j = \frac{|z_j|}{s\,\|W_j\|}, \tag{2}$$

where u_j is the output's *normalised margin*. A direction with a small cosine can still flip outputs whose margins are small, and even the row itself leaves unchanged the outputs whose margins exceed α.

Unless marked out of distribution (OOD), we measure steering in distribution at $\alpha = 2$, and we report an effect when it has the same sign in every matched pair (Appendix C).

3 The prediction target changes which readout row the probe aligns with most

What happens to a probe direction when only the readout's target changes? With data, initialisation and batch order fixed, we find that shifting the target moves the DiM direction for clear(x) away from the clear(x) row and toward the holding(x) row, while clear(x) stays decodable. We first compare Y_t with Y_{t+1} (Section 3.1) and ask whether the labels alone predict the move (Section 3.2). We then use Y_{t-1} to separate the labels' statistics from their uncertainty (Section 3.3).

3.1 What the next-state target changes

To see what the target does, we place each model in Figure 1b by two cosines of d, the DiM direction for clear(x). The horizontal axis is its cosine with its own readout row, the clear(x) row, and the vertical axis its cosine with the holding(x) row. Current-state models sit at the lower right, close to their own row and near zero on the holding(x) row. Every next-state model sits higher and further left, beside the previous-state models we return to below. In every pair the first cosine falls and the second rises, and d ends up aligned more with the holding(x) row than with its own (Table 1). The rotation also holds in three new seeds and in one pair continued to 100 epochs (Appendices C and D). It also holds under a cosine that no invertible linear change of the activation coordinates alters, with the readout adjusted to keep every logit: in all 12 shifted-target models d stays closer to the holding(x) row than to its own, 0.91–0.95 against 0.53–0.65 under the next-state target and 0.48–0.54 against 0.11–0.22 under the previous-state target (Appendix D).

The coupling of the labels suggests where the direction goes, and Section 3.2 tests this. As the worked episode of Section 2 shows, in random legal play clear(x) now is necessary for holding(x) one step later, so it is informative about that output. The own-row cosine stays positive in every pair, so row alignment is redistributed rather than reversed. Yet clear(x) stays decodable above 97% under the current- and next-state targets (Table 1). Decodable information need not be what a model uses (Elazar et al., 2021), and here the direction that decodes clear(x) has moved toward the row of a neighbouring fact.

Table 1: **Shifting the target moves d from its own readout row toward the holding(x) row.** Readout-row cosines of the DiM direction d for clear(x) with its own row and with the coupled holding(x) row: observed (means over three network seeds) and predicted by the UFM from the labels alone (medians over three UFM seeds). The last column is the state probe's held-out accuracy on the current-state clear predicates. Ranges and the state probe's cosines are in Table 3.

		DiM d		UFM		state probe
Target	Width	own	coupled	own	coupled	acc. (%)
current	64	0.85	−0.06	0.94	−0.03	99.9
next	64	0.24	0.59	0.40	0.59	99.5
previous	64	0.16	0.61	0.40	0.59	92.7
current	128	0.86	−0.05	0.93	−0.03	100.0
next	128	0.24	0.64	0.40	0.59	97.4
previous	128	0.28	0.58	0.39	0.60	97.7

Table 2: **Steering along four directions under the three targets.** Flip rate ρ (%) of the clear bits in distribution at $\alpha = 2$, as means over three seeds. At a matched norm the readout row's flip rate is the largest possible. Wider models have larger normalised margins (Table 9), so every direction, the row included, flips fewer bits. Table 7 gives the cosines and Table 8 the values per pair.

Target	Width	readout row W_j	DiM d	state probe w_{state}	output probe w_{output}
current	64	67.1	48.0	50.1	53.4
next	64	98.3	60.1	50.2	98.2
previous	64	60.2	0.5	0.2	51.4
current	128	16.5	6.7	8.6	8.7
next	128	87.3	33.5	36.9	85.7
previous	128	19.9	0.3	0.1	13.6

3.2 The labels predict where the direction goes

Is the rotation something the network does, or something the labels demand? To find out, we remove the network. Our unconstrained features model (UFM; after Mixon et al., 2022; Zhu et al., 2021) fits free per-example features at two adjacent positions and one shared affine readout to the same shifted labels. It is trained with per-bit binary cross-entropy and weight decay, as the networks are, with free features in place of the transformer. If the labels alone set the destination, the UFM's direction for clear(x) should also align more with the holding(x) row than with its own. It does, whereas with a same-position readout it stays close to its own row.

The UFM predicts the destination closely. The networks match its holding(x)-row cosine to within 0.07 in every pair, and in Figure 1b its ring for the shifted targets sits at the height of the shifted models. The ring sits further right, however. At both widths the networks keep less of their own row than the UFM predicts (Table 1). We read this shortfall as the network's own contribution.

3.3 A deterministic target: label statistics suffice

The move from Y_t to Y_{t+1} changed two things at once. It changed the labels' statistics, and because the next action is unseen, it made the labels uncertain given the prefix. To separate the two, we train a third model per pair that predicts the *previous* state Y_{t-1}, the state before the last observed action. Like Y_t, this target is an exact function of the prefix, and like Y_{t+1} it is displaced in time.

If uncertainty drove the rotation, the previous-state direction would stay near its own row. Instead, the DiM direction again aligns more with the holding(x) row than with its own, and its own-row cosine stays positive. Because in random legal play the hand alternates with position, labels one step *back* couple clear(x) to holding(x) as labels one step ahead do, and the UFM predicts the same destination when we refit it to them. It matches the holding(x)-row cosine to within 0.03 in every pair. As before, the network keeps less of its own row than the UFM predicts, and this shortfall halves from width 64 to width 128 (Table 1). The label statistics therefore suffice, and uncertainty is unnecessary.

The readout row, by contrast, follows the target. When we fit a probe to this model's own target, it lies close to the clear(x) row, with cosine at least 0.87. Under both deterministic targets the row also lies close to the DiM direction of the model's own labels, a property the uncertain next-state target lacks (Appendix B).

The state probe sits elsewhere. Fitted to the same labels as d, it is near-orthogonal to *both* rows. The readout's row space decodes the present fact no better than the labels Y_{t-1} alone do, and the rest of that information lies in the null space, where the state probe puts most of its squared norm (Appendix D). Thus the readout row tracks its training target, while d tracks the label statistics.

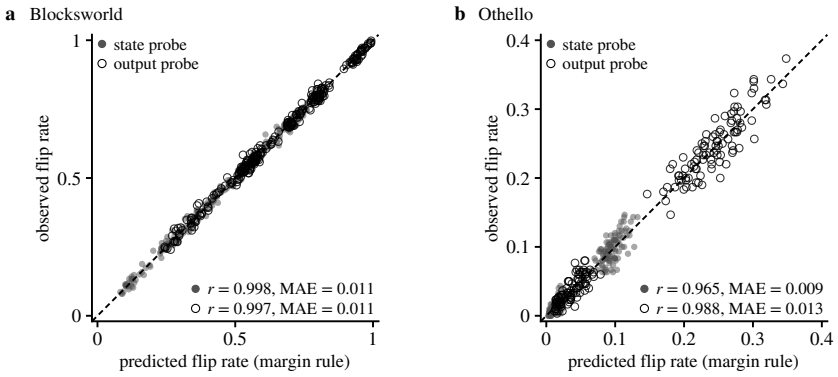

Figure 2: **At an affine readout, the margin rule reproduces observed flip rates.** Observed against predicted flip rate. The prediction is the margin rule of equation 2, which uses only the direct logit effect $W_j \cdot \delta$ and the clean margins, so points on the dashed identity line mean that the rule holds. Filled: state probe; open: output probe; r: Pearson correlation; MAE: mean absolute error. **(a)** Blocksworld, $\alpha = 2$, one point per model, output bit and probe. **(b)** Othello next-move models of a separate family (Appendix G), $\alpha = 1$, one point per model, square and probe. Random directions obey the same rule ($r = 0.973$).

4 Steering along the probe direction

4.1 The margin rule predicts steering

At an affine readout the margin rule of equation 2 is exact, so each flip rate can be computed from the unsteered model, from the direct logit effect $W_j^\top \delta$ and each output's clean margin. Figure 2a confirms this, up to sampling noise, over the 27 state-tracking models ($r = 0.998$ for the state probe, with the output probe as close). The Othello next-move readout is a softmax, where the rule is only approximate, yet it still predicts flip rates ($r = 0.965$ for the state probe, Figure 2b), for random directions too. Whether a direction steers therefore comes down to its cosine with the row and the margins it faces.

4.2 Why next- and previous-state models steer differently

The two shifted targets move d alike, yet they steer differently (Table 2), because their margins differ. Next-state outputs are uncertain, with a median normalised margin of 0.38 in the narrower models, and previous-state outputs are confident, with 1.82 (Table 9). Small next-state margins let d flip many clear outputs even after it has turned toward the holding(x) row. It keeps most of the fraction of the row's flip rate that it had under the current-state target, and even random directions flip some outputs. Previous-state margins are large, so the small component of d on the clear(x) row crosses almost none of them. There d keeps medians of only 0.007 and 0.010 of the row's flip rate at the two widths, and the state probe no more, as the margin rule predicts. The UFM's own-row cosine, combined with the same margins, predicts more of this steering than the networks show (Appendix E).

Out of distribution, where margins shrink, d recovers part of the row's flip rate at both widths (Table 9). Near-zero steering under the previous-state target is thus an in-distribution result, set by confident outputs. It is also the correct behaviour, since the scored output reports the state before the last action; an experimenter who read only the output's name would read it as the model not using clear$_t(x)$.

4.3 The output probe recovers the readout row

In the matched pairs, steering separates the two probes only under a shifted target. Under the current-state target the output probe and the state probe flip about as many clear bits, but under the next-state target the output probe flips 48 points more (Table 2). Fitted to the model's own output bits, the output probe w_{output} recovers the readout-aligned direction without access to the readout. It keeps

most of the row's flip rate in every previous-state model (Tables 2 and 8). Across the 27-model sweep, at matched norms, it flips 72% of clear bits and the state probe 39%, while the readout row, optimal for its own logit, is at least as good (Table 10). The output probe is therefore best used as a diagnostic that finds the readout's direction when the readout is hidden. Its advantage over the state probe combines row alignment with differences in estimator, errors and margins, and where the row is visible, steering along it is at least as effective.

4.4 Is the direction used?

Steering after the last block tests alignment with the readout, not whether the network uses the direction. To test use, we steer one block earlier, at the output of the first block, along the DiM direction of clear(x) computed there, and read the readout after the last block, which is nonlinear. We push clear(x) toward true on held-out examples where it is false, at $\alpha = 2$, and compare each output's change in logit with that of 50 norm-matched random directions, under criteria we fixed before the test split was read (Appendix E). Under the current-state target the direction drives its own output, with z scores of 19–31 against the random directions. Under both shifted targets it also drives the coupled holding(x) output, in the direction the label coupling predicts. It raises that logit by 2.0–2.3 under the next-state target and by 5.7–8.8 under the previous-state target, with z of 7–21 in all 12 models. Later computation therefore reads the direction, and under a shifted target its effect reaches the output that the target couples to it; under the next-state target it also moves its own output (z of 5–8, with 7.1–11.8% of its answers changed). The holding(x) shifts are in logits rather than in answers: the holding(x) logits start far below zero, and at most 2.2% of them cross it. We had also required that the previous-state clear(x) output, which reports the state before the last action, stay within the random band ($|z| < 2$). It changes at most 0.08% of its answers, but two of the three width-128 models exceed the band (up to $z = 2.2$), so this criterion is not met.

5 Beyond the primary encoding

We close with two tests of scope. The first removes the counting shortcut, and the second moves the target swap to Othello.

5.1 Without the counting shortcut

The counting shortcut could in principle produce the rotation on its own. In our standard encoding every action names the blocks it touches, so a model can read the state off action counts (Section 2). We therefore trained models with all three targets on an encoding that hides unstacking's source block, where the action counts leave the state undetermined. Under both shifted targets the DiM direction for clear(x) again aligns more with the holding(x) row than with its own, in all three seeds at width 64 (Table 11). On this encoding the rotation therefore appears without the counting shortcut.

The current-state models also solve episode pairs that no function of the action multiset can separate, a task beyond the counting shortcut. Under the previous-state target the output probe again outranks d, which keeps at most 0.09 of the row's flip rate (Appendix F).

5.2 Othello: the target sets the sign

We next ask whether the target orients the probe direction in a second domain, with the architecture and the games held fixed. An Othello sequence model's board is recoverable and causally editable (Li et al., 2023a), and linearly so with player-relative labels (Nanda et al., 2023). We instead vary only the target of the affine readout and train separate networks on identical games. The *occupancy readout* predicts the board's occupancy after the next move. The *next-move readout* predicts the next move played, as a 65-way softmax that includes the pass. In both we probe the current occupancy and take its DiM direction d.

The labels predict a sign. Occupancy is order-invariant. Apart from the four starting squares, a square is occupied if and only if it appears in the multiset of moves played, and the occupancy target adds only the next move's square. The next move is order-dependent, since its legality depends on the colour configuration, which the multiset does not fix. Under the occupancy readout, d for a square's

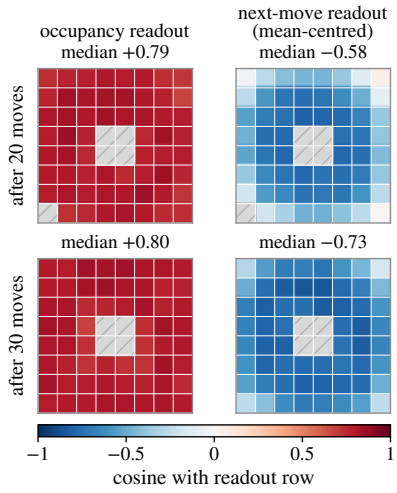

Figure 3: **In Othello, the readout's target sets the sign of the occupancy direction's alignment with its row.** Networks with the same architecture, trained separately on the same games, whose readout predicts the occupancy after the next move (left) or the next move (right), read after 20 and 30 moves. Each square shows the cosine of its occupancy DiM direction d with its readout row, averaged over 3 seeds, so red means alignment with the row and blue the opposite. Next-move rows are mean-centred. Titles: median over squares. Hatched: occupancy nearly constant, so no direction. The next-move readout predicts weakly (Appendix G), so we report geometry only.

occupancy should therefore lie along that square's row. Under the next-move readout it should lie against it, because an occupied square is never a legal move.

We find the predicted sign. Figure 3 colours each square by the cosine of its d with that square's readout row, and nearly every square is red under the occupancy readout and blue under the next-move readout. After 20 moves the median cosine is 0.79 under the occupancy readout and -0.58 under the next-move readout. We mean-centre the next-move rows because a softmax ignores a common shift of its rows. The sign holds in all 3 seeds, on freshly generated games and after 30 moves. In Othello, too, the readout's target sets whether the probe direction aligns with its row or against it.

6 Related work

Probe accuracy can reflect the probe's capacity (Hewitt & Liang, 2019), and decodable information need not be used (Belinkov, 2022; Elazar et al., 2021; Ravichander et al., 2021). The decodability–steerability gap itself is documented. Detection and control directions can be nearly orthogonal (Galeone et al., 2026), the best-read layer need not be the best-steer one (Wang et al., 2026), and patching can misidentify a causal subspace (Makelov et al., 2024). Closer to our view, unembedding projections predict where steering succeeds and trained probes do not (Billa, 2026), and a near-perfect count probe can sit nearly orthogonal to digit readout rows (Garcia, 2026). These studies keep the readout's target fixed. Only Vakalis (2026) swaps the target in matched models, with the output layer's form changing too, and measures latent recovery. Others steer the learned mechanism with intermediate training tasks such as state-parity prediction (Li et al., 2025), or add an auxiliary next-latent loss (Teoh et al., 2025). We vary only the readout's target and measure probe–readout geometry.

Multi-label and next-token neural collapse (Li et al., 2024; Ma et al., 2026; Zhao et al., 2024) describe label-driven readout geometry under losses other than our per-bit binary cross-entropy (Appendix B). DiM maximises a worst-case directional-derivative objective over admissible linear predictors (Belrose, 2023), recovers transferable truth directions (Marks & Tegmark, 2024), and underlies inference-time intervention, contrastive activation addition and refusal ablation (Li et al., 2023b; Panickssery et al., 2024; Arditi et al., 2024). Our fit of w_{output} to a model's own outputs has precedent in the tuned

lens's distillation objective (Belrose et al., 2023). In neuroimaging, the weights of a linear decoder, a filter, generally differ from the activation pattern of the signal it decodes, because the filter must also cancel other signals that share its channels (Haufe et al., 2014); the DiM direction is such a pattern and the readout row such a filter, and we vary only what the filter was trained to predict. Shallow transformers learn shortcuts rather than simulating the recurrence (Liu et al., 2023); our counting shortcut is a solution of this kind.

7 Discussion and limitations

What does a failed steering experiment tell us about a model? In our matched models, changing only the readout's target produces, in distribution, a controlled analogue of the decodability–steerability gap that observational studies report (Galeone et al., 2026; Billa, 2026). The fact stays decodable, but the probe follows the true state and the readout its target, and under the previous-state target the output we score reports the same fact one action earlier. Steered one block earlier under that target, the direction still moves the network's logits, but those of the coupled holding(x) output rather than the one we score (Section 4.4). In practice, then, a failed steering experiment calls for one check before any conclusion about the representation. We would ask what the readout was trained to predict and, if that differs from the probed fact, steer along the readout row instead, or compare the probe with one fitted to the model's own outputs when the readout is hidden (Section 4.3).

Limitations. Our models are small, with 6 matched pairs from three independent seeds and 3 Othello seeds, so we report sign agreement rather than intervals (Appendix C). In a further test with an always-legal no-op (Appendix H), forward targets align d with the holding(x) row beyond the range of label-only fits in all 12 models, but a backward control we pre-specified failed its criterion, so we cannot tie this excess to the forward direction. The primary encoding permits an action-count solution, and we leave open whether the networks update the state step by step or hold a world model. Without that shortcut, our evidence is one encoding, one width and three seeds, and its accuracy threshold was fixed after the first seed, so only the other two seeds test it (Appendix F). Behind a nonlinearity, where equation 1 does not hold, we have only a fine-tuned base Qwen2.5-0.5B (Qwen Team, 2024). On one checkpoint, at its validation-selected strength and matched norms, its output probe steers more than the state probe but less than two label-free directions (Table 13). In matched low-rank adaptation (LoRA) fine-tunes, the predicted steering drop appears in only 2 of 3 pairs, missing its pre-specified criterion (Appendix I).

8 Conclusion

We asked whether a probe's failure to steer can depend on what the model's readout was trained to predict. In matched models that differ only in that target, the DiM direction of clear(x) aligns more with the holding(x) readout row than with its own under a shifted target, while clear(x) stays decodable. The outputs' margins then decide how much steering along the direction survives. In Othello, the target sets the sign of the occupancy direction's alignment with its own row. The DiM direction's alignment with the readout thus reflects what the readout was trained to predict as well as what the representation encodes.

Reproducibility statement

The supplementary material contains the code, the result files and the scripts that regenerate every figure and table, with a README mapping each to its script. It also contains the pre-specified criteria and their protocols. Table 15 lists each pre-specified criterion and what it showed, and marks the post hoc and exploratory analyses. Appendix A gives the seeds, grids, training details, matched intervention norms and validation-only tuning. We will release a public repository at camera-ready.

AI use statement

The authors designed the research question, chose the claims, verified every technical step and take full responsibility for the content. Under their direction, LLM-based assistants helped refine hypothe-

ses and draft the pre-specified criteria, draft derivations (checked by hand), write and independently check the experiment, figure and number-extraction code, draft readings of the results and parts of the text, search and summarise related work (every citation checked by the authors), and critique drafts. Every number is computed from released result files by released scripts; no LLM generated data, acted as an annotator or judge, or is an author.

References

Alessandro Achille, Matteo Rovere, and Stefano Soatto. Critical learning periods in deep networks. In *International Conference on Learning Representations (ICLR)*, 2019.

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

Nora Belrose. Diff-in-means concept editing is worst-case optimal. EleutherAI Blog, December 2023. URL `https://blog.eleuther.ai/diff-in-means/`.

Nora Belrose, Igor Ostrovsky, Lev McKinney, Zach Furman, Logan Smith, Danny Halawi, Stella Biderman, and Jacob Steinhardt. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*, 2023.

Jayadev Billa. Predicting where steering vectors succeed. *arXiv preprint arXiv:2604.15557*, 2026.

Hiskias Dingeto. Train the model, not the reader: Decodability supervision for verifiable activation explanations. *arXiv preprint arXiv:2607.20379*, 2026.

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. Amnesic probing: Behavioral explanation with amnesic counterfactuals. *Transactions of the Association for Computational Linguistics*, 9:160–175, 2021.

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

Gabriel Garcia. The right answer, the wrong direction: Why transformers fail at counting and how to fix it. *arXiv preprint arXiv:2605.03258*, 2026.

Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Time matters in regularizing deep networks: Weight decay and data augmentation affect early learning dynamics, matter little near convergence. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Stefan Haufe, Frank Meinecke, Kai Görgen, Sven Dähne, John-Dylan Haynes, Benjamin Blankertz, and Felix Bießmann. On the interpretation of weight vectors of linear models in multivariate neuroimaging. *NeuroImage*, 87:96–110, 2014. doi: 10.1016/j.neuroimage.2013.10.067.

John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019.

Erik Jenner, Shreyas Kapur, Vasil Georgiev, Cameron Allen, Scott Emmons, and Stuart Russell. Evidence of learned look-ahead in a chess-playing neural network. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Belinda Z. Li, Zifan Carl Guo, and Jacob Andreas. (How) do language models track state? In *International Conference on Machine Learning (ICML)*, 2025.

Kenneth Li, Aspen K. Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Emergent world representations: Exploring a sequence model trained on a synthetic task. In *International Conference on Learning Representations (ICLR)*, 2023a.

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

Pengyu Li, Xiao Li, Yutong Wang, and Qing Qu. Neural collapse in multi-label learning with pick-all-label loss. In *International Conference on Machine Learning (ICML)*, 2024.

Bingbin Liu, Jordan T. Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Transformers learn shortcuts to automata. In *International Conference on Learning Representations (ICLR)*, 2023.

Xiaoxuan Ma, Yixuan Yang, Song Li, and Xiangyun Hui. How label imbalance shapes geometry: A general spectral analysis of multi-label neural collapse. *arXiv preprint arXiv:2605.01897*, 2026.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018.

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

Samuel Marks and Max Tegmark. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. In *Conference on Language Modeling (COLM)*, 2024.

Tianyi Men, Pengfei Cao, Zhuoran Jin, Yubo Chen, Kang Liu, and Jun Zhao. Unlocking the future: Exploring look-ahead planning mechanistic interpretability in large language models. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024. doi: 10.18653/v1/2024.emnlp-main.440.

Dustin G. Mixon, Hans Parshall, and Jianzong Pi. Neural collapse with unconstrained features. *Sampling Theory, Signal Processing, and Data Analysis*, 20(2):11, 2022. doi: 10.1007/s43670-022-00027-5.

Neel Nanda, Andrew Lee, and Martin Wattenberg. Emergent linear representations in world models of self-supervised sequence models. In *Workshop on Analyzing and Interpreting Neural Networks for NLP (BlackboxNLP)*, 2023.

Nina Panickssery, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. Steering Llama 2 via contrastive activation addition. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024. doi: 10.18653/v1/2024.acl-long.828.

Vardan Papyan, X. Y. Han, and David L. Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663, 2020. doi: 10.1073/pnas.2015509117.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In *International Conference on Machine Learning (ICML)*, 2024.

Kiho Park, Yo Joong Choe, Yibo Jiang, and Victor Veitch. The geometry of categorical and hierarchical concepts in large language models. In *International Conference on Learning Representations (ICLR)*, 2025.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Abhilasha Ravichander, Yonatan Belinkov, and Eduard Hovy. Probing the probing paradigm: Does probing accuracy entail task relevance? In *Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2021.

Jayden Teoh, Manan Tomar, Kwangjun Ahn, Edward S. Hu, Tim Pearce, Pratyusha Sharma, Akshay Krishnamurthy, Riashat Islam, Alex Lamb, and John Langford. Next-latent prediction transformers learn compact world models. *arXiv preprint arXiv:2511.05963*, 2025.

Curt Tigges, Oskar J. Hollinsworth, Atticus Geiger, and Neel Nanda. Language models linearly represent sentiment. In *Workshop on Analyzing and Interpreting Neural Networks for NLP (BlackboxNLP)*, 2024. doi: 10.18653/v1/2024.blackboxnlp-1.5.

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

Donna Vakalis. What a world model represents is three questions. *arXiv preprint arXiv:2607.06640*, 2026.

Yibo Wang, Jisheng Dang, Bimei Wang, Yitao Wu, Wencan Zhang, Hong Peng, Jizhao Liu, Bin Hu, Qi Tian, and Tat-Seng Chua. Read-best is not steer-best: A probing–steering layer dissociation in omni-modal large language models. *arXiv preprint arXiv:2609.22135*, 2026.

Yize Zhao, Tina Behnia, Vala Vakilian, and Christos Thrampoulidis. Implicit geometry of next-token prediction: From language sparsity patterns to model representations. In *Conference on Language Modeling (COLM)*, 2024.

Zhihui Zhu, Tianyu Ding, Jinxin Zhou, Xiao Li, Chong You, Jeremias Sulam, and Qing Qu. A geometric analysis of neural collapse with unconstrained features. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

A Models, data and probes

The appendices follow the main text: models, task and statistics (A–C), geometry and steering (D, E), the encoding without the counting shortcut and Othello (F, G), the forward excess and the language model (H, I), additional experiments (J) and the pre-specified criteria (K).

Within a matched pair only the label differs, and every family uses the same estimators, with the scale, norm matching and strength grid given for it below.

State-tracking models (Sections 3 and 4). These are pre-norm transformers of depth 2 and width $64/128$ with 4 attention heads, head dimension equal to the width divided by the number of heads, MLP width $4\times$ the model width, GELU activations and LayerNorm, a 53-token action vocabulary and context 60. No final LayerNorm sits on the readout's path, so the readout reads the residual stream after the last block directly. The state readout is a single affine map from the width to the 36 output bits, trained with a per-bit binary cross-entropy. We read activations, and inject δ, at the residual stream after the last block, with nothing between that site and the readout.

Training uses AdamW at learning rate 0.001, weight decay 0.01, betas $(0.9,\ 0.98)$ and batch size 128, with 300 warmup steps followed by a cosine decay whose horizon is 100 epochs. All 12 primary models stop at a fixed 30 epochs, a stopping point we chose on validation data before examining the final splits. A pair continued on the same schedule to 100 epochs gives the sensitivity row of Table 6. The grid is widths $64/128\times$ seeds 1–3 $\times$ the two targets. The six previous-state models of Section 3.3 reach at least 0.9993 complete-state validation accuracy, with all 36 bits exact, against a pre-specified threshold of 0.99. Within a pair, the initial parameters, training episodes, minibatch order, input positions, mask, optimiser, schedule and update count are identical. The models of a triple consume the same input window and differ only in whether the label at a position is the state that holds there, one action later or one action earlier.

Data and leakage control. Episodes start with all blocks on the table and select uniformly from the sorted legal actions, which we call random legal play. Separate random seeds generate the model-training, model-validation, probe-fitting and probe-validation splits and the two final evaluation splits. We use $4{,}096$ training and 512 validation episodes of uniform length 10–30, and per evaluation regime $2{,}000$ probe-fitting, 500 probe-validation and $1{,}000$ final sequences. The in-distribution (ID) regime has length 30 and is read at position 28. The out-of-distribution (OOD) regime has length 50 and is read at position 40. Generation rejects any episode whose complete sequence, or whose input prefix up to the read position, already occurs in another split, including the fixed episode sets on which the sweep models below were trained and validated.

Probe fitting and strength grids. We fit every logistic probe direction on frozen activations, selecting the inverse regularisation constant from $\{0.01,\ 0.1,\ 1,\ 10\}$ on a disjoint validation split, with ties going to the smaller value. The DiM direction has no fitted constant. The primary regime is in distribution, at position 28. Out of distribution, at position 40, the estimand is OOD transfer. There we apply directions, constants and the norm-calibration scalar fitted in distribution unchanged, and we report the result beside every in-distribution steering number that it changes. The activation scale is $s = \sqrt{\text{mean}(\text{var}(h))}$, the root mean of the per-coordinate variances of the fitting-split activations, so that between-feature mean offsets do not enter. Every direction is unit-normalised and given the same perturbation norm $\alpha \cdot s$ over the strength grid $\{0.5,\ 1,\ 2,\ 4\}$, with $\alpha = 2$ primary. We discard any stratum whose realised norm is more than 1% from the requested norm and list it as an exclusion. Geometry is quoted for the DiM direction, and steering for the DiM direction and the state probe, and each comparison stays within one estimator.

Where the two probes coincide. Where a model's output bits coincide with the truth, w_{state} and w_{output} are fitted to the same labels, and a small difference between them is definitional. We flag that case wherever it arises. Cosines are raw at the per-bit sigmoid state readout and mean-centred at the softmax next-move readout of Section 5 (Appendix G).

State-tracking sweep (Section 4.3, Table 10). The sweep has 27 models, over depths 2, 4 and 6, widths 64, 128 and 256 and seeds 1–3. They share the architecture, readout and read site of the primary models, and their readout is trained on the next state. They were trained on a fixed set of $16{,}004$ episodes with a masked per-bit binary cross-entropy, the AdamW settings and warmup given above, batch size 256 and a cosine decay with a horizon of 200 epochs, and stopped early on held-out bit accuracy with a patience of 30 epochs. For the matched-norm steering comparison of Table 10, we probe and steer them on fresh splits generated as above, with the same sizes, positions and grids.

Othello models (Section 5). These are transformer encoders with a causal attention mask and learned token and position embeddings, 4 layers of width 128 with 4 attention heads and a feed-forward width of $4\times$ the model width. We train them for 30 epochs on $8{,}000$ random-legal games with AdamW at learning rate 0.001, weight decay 0.01 and batch size 64, at 3 seeds per readout. The two readouts share these inputs and this recipe and differ only in the affine readout and its loss. The occupancy readout has 64 bits and is trained with binary cross-entropy on the board's occupancy after the next move. The next-move readout is a 65-way softmax over the 64 squares and a pass, trained with cross-entropy on the next move played. The occupancy probes are fitted to the current occupancy. We read activations, and inject δ, at the output of the last encoder block, which the readout reads directly. Probe fitting, moment calibration and evaluation use 300, 300 and 500 freshly generated random-legal games under separate seeds, none of which repeats an exact prefix of the models' training or evaluation games. Position 20 is primary and 30 secondary, and interventions use the same activation scale over the strength grid $\{1,\ 2,\ 4\}$.

Fine-tuned language model (Section 7, Appendix I). We use a checkpoint of the base Qwen2.5-0.5B model (Qwen Team, 2024), fine-tuned with low-rank adaptation (LoRA) on this same Blocksworld task. We read it at the answer slot of position 21 on freshly generated episodes, with 320 fitting, 80 validation and 120 final sequences, none of which repeats an exact prompt prefix of those fixed episode sets. Candidate injection sites are the zero-based decoder-block outputs $L18$, $L22$, $L23$, taken before the final RMSNorm. We selected the reported site $L22$ and the steering strength $\alpha = 4.0$ on the validation split, and we match norms on the realised per-dimension RMS. A nonlinearity separates that site from the unembedding, so equation 1 does not govern it.

B What the task alone identifies

The labels predict where the direction goes, to within the stated residuals, but they do not identify how the network gets there. A model of the labels alone, with no trained network (the UFM of Section 3.2), reproduces the sign and the coarse size of the geometry change. The task fixes the affine part that equation 1 acts on, but it leaves norms, variance shares and raw cosines free. Rescaling the hidden coordinates by an invertible diagonal matrix, and undoing the rescaling in the readout, changes all three while leaving every logit unchanged. We therefore quote raw readout-row cosines at the per-bit

sigmoid state readout, mean-centre them only at the softmax next-move readout (Appendix G), and treat the UFM as a model of the label statistics rather than as the generative account of the geometry.

A simpler predictor makes the same point. We forecast the *steering gain*, the change in the clear logit per unit of activation scale when steering along d, in two ways from the labels. One factorises the matrix of the labels' Bayes logits into features and readout with balanced singular values, the form of the UFM and of neural collapse. The other uses the label logits themselves as features, with no factorisation. On six exploratory models, the second predicts the gain at least as well, with a mean absolute error before calibration of 0.079 against 0.149. We then searched, with a pre-specified criterion, over 625 auxiliary-target settings for a design on which the two predictors disagree by at least 0.05. The search found 0 such settings, where 3 were required, so we have no design that tells the two predictors apart.

Relation to neural-collapse results. Under the balance assumptions of Li et al. (2024), pick-all-label UFM minima have a simplex equiangular tight frame (ETF) as classifier and feature means proportional to tag-wise sums of classifier rows. Under imbalance, Ma et al. (2026) describe count-weighted sums of rescaled classifier directions and covariance-dependent stability. Under next-token cross-entropy, contexts with identical next-token supports converge to the same limiting direction (Zhao et al., 2024). Each of these losses differs from our per-bit binary cross-entropy, so we apply them to d under a shifted target only as an analogy. One neural-collapse property can be checked directly: self-duality (NC3; Papyan et al., 2020), whose per-bit analogue puts each readout row along the DiM direction of the model's own labels. At width 64 it holds for the deterministic targets, with cosines 0.85 under the current-state and 0.89 under the previous-state target, and fails for the uncertain next-state target (0.41).

C Statistical treatment

Our Blocksworld claims rest on sign agreement across three independent seeds, with every direction fitted away from the cases that score it. The unit of analysis is the matched pair in the state-tracking experiments and the seed in Othello, with 6 pairs across two widths and three initialisations and 3 seeds per Othello readout. Predicates, positions and evaluation cases within a model are repeated measures on that unit, and we aggregate them before any comparison across units. The two widths within a seed share the training episodes and the minibatch order and differ in width and initialisation, so the 6 pairs come from three independent seeds. At this n we report agreement in sign and ranges over the units rather than intervals, because a bootstrap over three seeds has no useful coverage. A resampling range, where we quote one, is a range over the paired units. A count such as 6 of 6 states consistency across pairs, and we attach no significance test to it. At the seed level, 6 of 6 pairs is three of three seeds.

Six seeds. The geometry also holds beyond three seeds. A further sweep trains the three targets at widths 64 and 128 over six seeds, giving 12 matched triples. Seeds 1–3 reproduce the primary models with identical initialisation and batch order, so they are repetitions rather than new evidence. Seeds 4–6 are new, with 18 models. All seeds share one data draw and the two widths share batch order within a seed, so the independent units are seeds. The pre-specified geometry criterion asks for an own-row drop with a coupled-row rise on the development split. It holds for both shifted targets at both widths, with the same signs in 6 of 6 seeds and in 3 of 3 new seeds. Beyond the criterion, the previous-state DiM direction keeps 0.007 and 0.010 of the row's flip rate at $\alpha = 2$ in distribution over the six seeds, and 0.007 and 0.009 over the new seeds alone.

Fitting and evaluation splits. We fit directions, regularisation constants and calibration scalars on splits disjoint from the evaluation splits, and we fixed each analysis before examining its final evaluation split. The exception is the exploratory read of Appendix J.1, which is tested on development data and reported only as exploratory. Development results are not independent of one another (Appendix K).

D WHERE THE PROBE DIRECTION GOES

The rotation of Section 3 holds in three further seeds and in one pair continued to 100 epochs, independently trained models in an exploratory run meet its cosine criteria, and in the shared-initialisation frame it is the probe direction that turns. Table 3 gives the geometry per width and target. Our unit of analysis is the matched pair (Appendix C), so Table 4 gives each pair's values, from which Section 3 reads its sign agreements.

Table 3: **Geometry of the probe direction under three prediction targets.** Mean over each width/target cell's three seeds, [minimum, maximum] beneath. Cosines of the probe direction for the current-state fact clear(x) with its *own* readout row and with the row of the *coupled* fact holding(x), for both estimators; the per-bit sigmoid readout needs no centring. *UFM* is the unconstrained features model's prediction, one value per width and target, and *residual* the observed DiM own-row cosine minus it (negative where the network keeps less of its own row than the UFM predicts). Bold: the DiM direction under the next- and previous-state targets. *Probe acc.* is the state probe's accuracy on the *current* state under every target, so the column is comparable across targets. Per-pair rows, Table 4.

	DiM d		state probe w_{state}		UFM			
Width, target	own	coupled	own	coupled	own	coupled	residual	probe acc. (%)
64, current	0.854	−0.062	0.868	−0.086	0.941	−0.034	−0.087	99.9
	[0.843, 0.866]	[−0.071, −0.047]	[0.850, 0.900]	[−0.143, −0.043]				[99.9, 99.9]
64, next	**0.244**	**0.587**	0.197	0.711	0.400	0.590	−0.157	99.5
	[0.237, 0.250]	[0.582, 0.590]	[0.179, 0.230]	[0.680, 0.743]				[99.4, 99.6]
64, previous	**0.161**	**0.610**	0.093	0.055	0.396	0.593	−0.234	92.7
	[0.128, 0.189]	[0.602, 0.618]	[0.081, 0.104]	[0.019, 0.105]				[91.3, 93.8]
128, current	0.856	−0.049	0.889	−0.027	0.932	−0.028	−0.077	100.0
	[0.849, 0.862]	[−0.058, −0.043]	[0.885, 0.895]	[−0.046, 0.009]				[100.0, 100.0]
128, next	**0.240**	**0.645**	0.266	0.460	0.399	0.594	−0.159	97.4
	[0.238, 0.242]	[0.632, 0.659]	[0.251, 0.287]	[0.400, 0.510]				[97.2, 97.4]
128, previous	**0.276**	**0.580**	0.050	0.094	0.394	0.597	−0.118	97.7
	[0.268, 0.280]	[0.569, 0.589]	[0.031, 0.076]	[0.073, 0.137]				[96.6, 98.4]

Which object moves? This comparison is post hoc. Between matched models, the change is concentrated in the angle between clear(x)'s DiM direction and the readout rows. Its cosines with the clear(x) and holding(x) rows change by 0.56–0.72, against at most 0.12 between seeds with the same target. The cosines among the DiM directions and among the rows change no more than between such seeds, on average at most 0.08 and 0.06 against up to 0.09 and 0.07. These comparisons need no common frame, so the change is a relative reorientation, and which object moves depends on the frame.

Matched models share their initialisation. In these shared coordinates, their clear(x) and holding(x) readout rows stay close to one another, with cosine at least 0.82, and to their initial values, at least 0.84. So does the DiM direction of on_table(x), at least 0.92. The DiM direction of clear(x) does not stay (0.41–0.57). In this frame the probe direction rotates.

Aligning the models by their activations reverses the attribution. In a frame aligned by an orthogonal map fitted to paired activations, d stays, with cosine at least 0.96, and the clear(x) row moves, with cosine at most 0.35 against at least 0.74 between seeds. That map, however, explains less variance between matched models (R^2 0.38–0.74) than between seeds with the current target, at least 0.78. Next-state seeds at width 128 reach only 0.67, below the matched width-128 pairs at 0.69–0.70. The map also carries d partly by construction, since d is a difference of means of the activations it aligns. We therefore read the rotation in the shared-initialisation frame. The turn also leaves the plane of the two rows. The share of d's squared norm in that plane falls from at least 0.72 under the current-state target to at most 0.50 under the shifted ones.

Table 4: **Per-pair geometry behind Table 3.** One block per matched pair, identified by width and seed; columns and conventions are those of Table 3, with no aggregation: *own* is the clear(x) readout row, *coupled* the holding(x) row, and *probe acc.* the state probe's accuracy on the *current* state under every target. The three models of a block share initialisation, training episodes, minibatch order and update count and differ only in the readout's prediction target.

		DiM d		state probe w_{state}		UFM			
Pair	Target	own	coupled	own	coupled	own	coupled	residual	probe acc. (%)
64, seed 1	current	0.866	–0.069	0.850	–0.071	0.941	–0.034	–0.075	99.9
	next	0.237	0.582	0.181	0.708	0.400	0.590	–0.163	99.5
	previous	0.167	0.618	0.104	0.105	0.396	0.593	–0.229	93.1
64, seed 2	current	0.852	–0.071	0.900	–0.143	0.941	–0.034	–0.089	99.9
	next	0.250	0.590	0.179	0.743	0.400	0.590	–0.150	99.6
	previous	0.189	0.610	0.081	0.041	0.396	0.593	–0.207	93.8
64, seed 3	current	0.843	–0.047	0.853	–0.043	0.941	–0.034	–0.098	99.9
	next	0.243	0.588	0.230	0.680	0.400	0.590	–0.157	99.4
	previous	0.128	0.602	0.096	0.019	0.396	0.593	–0.267	91.3
128, seed 1	current	0.849	–0.058	0.885	–0.044	0.932	–0.028	–0.084	100.0
	next	0.238	0.632	0.251	0.400	0.399	0.594	–0.161	97.4
	previous	0.280	0.569	0.031	0.073	0.394	0.597	–0.113	98.3
128, seed 2	current	0.862	–0.047	0.895	–0.046	0.932	–0.028	–0.071	100.0
	next	0.242	0.659	0.261	0.510	0.399	0.594	–0.157	97.2
	previous	0.280	0.582	0.043	0.073	0.394	0.597	–0.113	98.4
128, seed 3	current	0.857	–0.043	0.888	0.009	0.932	–0.028	–0.075	100.0
	next	0.239	0.644	0.287	0.469	0.399	0.594	–0.160	97.4
	previous	0.268	0.589	0.076	0.137	0.394	0.597	–0.126	96.6

Table 5: **Reading the current state from the readout's row space and from its null space** (Section 3.3). A logistic probe for the current-state clear predicates is fitted on the 36-dimensional readout row space span$(W^\top)$, on its null space and on the full activation (held-out accuracy, mean over three seeds). R^2: each part regressed on the last action, comparable for both parts. Row-space energy: fraction of the squared norm of w_{state} and of d in the row space (median; random direction: 0.56 and 0.28). Reading clear$_t$ from Y_{t-1} alone scores 0.870.

		probe accuracy			last-action R^2			row-space energy	
Width, target	null dim	row space	null	full	row space	null	full	w_{state}	d
64, current	28	0.999	0.962	0.999	0.389	0.395	0.389	—	—
64, next	28	0.995	0.927	0.995	0.206	0.291	0.212	—	—
64, previous	28	0.847	0.812	0.927	0.439	0.467	0.439	0.29	0.97
128, current	92	1.000	1.000	1.000	0.329	0.350	0.330	—	—
128, next	92	0.967	0.948	0.974	0.186	0.214	0.190	—	—
128, previous	92	0.866	0.956	0.977	0.376	0.391	0.376	0.13	0.95

A reparametrisation-invariant cosine. Raw cosines change under an invertible rescaling of the activations (Appendix B). For an invertible map $h \mapsto Ah$, with the readout adjusted so that every logit is unchanged, the DiM direction becomes Ad, a readout row $A^{-\top}w$ and the activation covariance $A\Sigma A^\top$, so $d^\top w / \sqrt{(d^\top \Sigma^{-1} d)(w^\top \Sigma w)}$ is invariant (exactly so without the ridge). We computed it at the readout site on the test split, with a ridge of $10^{-6}\,\mathrm{tr}(\Sigma)$ divided by the width, under criteria fixed before that split was read (Table 15). The DiM direction of clear(x) is closer to the holding(x) row than to its own in 12 of 12 shifted-target models and in 60 of 60 (model, x) cases, its own-row value stays positive, and under the current-state target it lies almost on its own row (0.98–1.00, against -0.37 to -0.24 for the holding(x) row). The same holds with ridges of 10^{-4} and 10^{-2} and with d and Σ from the fitting split.

What the row space carries. Under the previous-state target, the readout's row space holds the present no better than the previous labels do. Read from the 36-dimensional row space alone, the current-state fact scores medians of 0.845 and 0.868 at widths 64 and 128. That is at or below the accuracy of reading clear_t from the true previous-state labels Y_{t-1} alone, where a logistic read under the same fitting protocol scores 0.870, or 1.000 with the last action added. The row space therefore carries the previous state, which predicts the present only at that level. The remaining information about the present sits in the null space, with median accuracies of 0.809 and 0.965 at the two widths, against 0.931 and 0.983 for the full activation (Table 5 gives the seed means).

The state probe and the DiM direction sit on opposite sides of this split. The state probe places fractions of only 0.29 and 0.13 of its squared norm in the row space, against 0.56 and 0.28 for a random direction, so most of it lies in the null space. The DiM direction instead places fractions 0.97 and 0.95 of its squared norm in the row space, on other facts' rows. A probe refitted inside the row space sits 82–87° off its own readout row.

Training length and an independent replication. The next-state geometry does not look like a model trained worse or an artefact of where training stopped. The standard-encoding models of an earlier, independently trained and exploratory run (Appendix J.1) fall 1.01 and 0.96 points short of the Bayes-optimal accuracy at their two training lengths. On development data they meet the cosine criteria in all 3 seeds at both 60 and 90 epochs (Table 6), and the rotation also holds in the primary pair continued to 100 epochs. The primary next-state models' own next-state probe approaches that bit's Bayes-optimal accuracy, 68% against 72%, so we do not compare it with current-state decoding.

Table 6: **Robustness of the target-dependent geometry across training length and an independently trained replication.** Units of Table 3, except that probe accuracies here are read against each model's own target; the raw gap is $\rho(w_{\text{output}}) - \rho(w_{\text{state}})$. The first row continues one of the six primary matched pairs (width 64, seed 1) to 100 epochs; its entries are *current / next* target. The last two rows are the standard-encoding models of the independently trained replication (exploratory; Appendix J.1), at two training lengths, and give one value per column: the own-row cosine under the current-state target and the coupled-row cosine under the next-state target, the two cosines their criteria score.

	own-row cosine	coupled-row cosine	Probe acc. (%)	raw gap (pp)
100 epochs, width 64, seed 1	0.887 / 0.247	−0.052 / 0.665	100.00 / 68.42	0.00 / 32.80
independently trained standard-encoding models, cosine criteria, mean over 3 seeds				
60 epochs	0.882	0.662	—	—
90 epochs	0.884	0.681	—	—

E How much the directions steer

At the affine readouts, a direction steers an output through its component on that output's row, measured against the output's margins, and the margin rule accounts for every flip rate we measured there. Table 7 gives geometry and steering for the same four directions, and Table 8 each pair's values, from which Section 4 reads its sign agreements.

Sampled flip rates. Sampled flip rates agree with the expectation of the margin rule (Section 4.1) within binomial noise, and exactly on shared examples (Table 7).

The readout row and a timing control. The readout's own row bounds both models of a pair, flipping 42% and 93%, as equation 1 requires. The output probe's advantage is more than a matter of timing. As a timing control, we fit a next-state probe w_{next} to the state one position later. It flips few bits under the current-state target, 0.09%, and 72% under the next-state target, where its labels are the readout's own target. Aligning a probe's target with the readout's thus moves it part of the way to the readout row, and fitting the model's own output bits covers the rest. The output bits are linearly separable in every model, so label noise is not isolated as the cause of the output probe's alignment.

Table 7: **Geometry and steering of the same four directions, under the three prediction targets.** Clear predicates, in distribution, steering strength $\alpha = 2$; the same six width/target cells as Table 3, each a mean over three seeds. *own* and *coup.* are cosines with the fact's own readout row and with the coupled fact's row; ρ is the measured flip rate (%) from an end-to-end forward pass, the readout row's own being the maximum equation 1 sets. The margin rule of Section 4.1 is not a separate column: over all 96 model–direction pairs checked, which also include the next- and previous-state probes not shown here, the two agree to the printed precision, as expected at an affine readout. Per-pair rows, the random-direction control and each direction's flip rate as a fraction of the row's (relative flip rate): Table 8.

	state probe w_{state}			DiM d			output probe w_{output}			readout row W_j		
Width, target	own	coup.	ρ	own	coup.	ρ	own	coup.	ρ	own	coup.	ρ
64, current	0.87	−0.09	50.1	0.85	−0.06	48.0	0.89	−0.11	53.4	1.00	−0.21	67.1
64, next	0.20	0.71	50.2	0.24	0.59	60.1	0.99	0.12	98.2	1.00	0.12	98.3
64, previous	0.09	0.06	0.2	0.16	0.61	0.5	0.92	−0.18	51.4	1.00	−0.21	60.2
128, current	0.89	−0.03	8.6	0.86	−0.05	6.7	0.89	−0.03	8.7	1.00	−0.14	16.5
128, next	0.27	0.46	36.9	0.24	0.64	33.5	0.95	−0.01	85.7	1.00	−0.02	87.3
128, previous	0.05	0.09	0.1	0.28	0.58	0.3	0.91	−0.16	13.6	1.00	−0.15	19.9

Spread of the output–state gap. Under the next-state target, the output–state gaps of the 6 primary pairs span 39–53 points. One pair (width 64, seed 1), continued to 100 epochs, falls by 19 points, against a spread of 13 across seeds at the fixed steering strength. One pair at one later checkpoint cannot distinguish a trend from seed variation. Under the current-state target the gap is near zero wherever the fit is near-perfect, at 0.06, 0.14 and 0.00 points in the width-128 pairs, and slightly above zero where it is not (5.24, 1.26, 3.46). The gap is small where the fit is near-perfect, though not zero by construction, and Appendix J.1 shows an encoding where the current-target fit is imperfect. Resampling the 6 width/seed pairs yields descriptive ranges conditional on this grid rather than additional independent seeds.

Fractions of the row's flip rate. On an output bit, a direction buys its component on *that bit's* row, scaled by the margins the row must cross. At width 64 the DiM direction keeps 0.68–0.76 of the readout row's flip rate under the current-state target and 0.59–0.63 under the next-state one. At width 128 it keeps 0.38–0.43 against 0.37–0.40. Alignment with the coupled row buys nothing on the clear outputs. The previous-state DiM direction has cosine 0.56–0.62 with the holding(x) row and still moves almost no clear bits in distribution (Table 9). We use the DiM direction because mass-mean and inference-time-intervention directions are the established way to steer (Marks & Tegmark, 2024; Li et al., 2023b; Belrose, 2023).

The previous-state target. Under the previous-state target, in distribution at $\alpha = 2$, the DiM direction keeps 0.006–0.026 of the row's flip rate per model, and the state probe 0.000–0.007. Under OOD transfer at the same strength, the state probe flips 1.8–7.0% of clear bits and the output probe 37.7–62.7%. The probe fitted to these models' own target has own-row cosine 0.875–0.938 (Section 3.3). The output we score reports the previous state, not the probed one. The output named clear(x) agrees with clear$_{t-1}(x)$ in 99.9% of cases and with clear$_t(x)$ in 67%, as often as the two labels agree with each other. In the narrower models, in distribution, d flips at most 3.4% of clear bits at every tested strength, and random directions at most 0.68%, as the margin rule predicts. Raw output–state gaps under this target differ severalfold between widths. Normalising by the row's flip rate shrinks this difference without removing it, and both measures order the widths alike (Table 8).

Margins and the UFM's steering prediction. We computed Table 9, and the main-text quantities that cite it, post hoc from the saved activations, weights and result files. The margin rule reproduces every measured flip rate in the table exactly. The UFM's geometry, by contrast, predicts more steering than the networks show. At $\alpha = 4$, the UFM's own-row cosine combined with each model's margins predicts that the previous-state DiM direction keeps 0.34 and 0.065 of the row's flip rate at widths 64 and 128. The networks keep 0.025 and 0.017 in distribution. So the UFM predicts where d

Table 8: **Per-pair steering behind Table 7**, at $\alpha = 2$ on the clear predicates in distribution, one pair at a time, with the random-direction control. *Relative* divides a direction's flip rate by the readout row's; *rel. gap* is $[\rho(w_{\text{output}}) - \rho(w_{\text{state}})]/\rho(W_j)$, and *raw gap* is $\rho(w_{\text{output}}) - \rho(w_{\text{state}})$ in points, which depends on the margin scale. Dashes, here and in later tables: not measured. The random-direction control was measured for the previous-state models; Table 9 gives it for every target.

		ρ (%)					relative			
Pair	Target	w_{state}	DiM d	w_{output}	W_j	random	w_{state}	w_{output}	rel. gap	raw gap (pp)
64, seed 1	current	47.4	49.7	52.6	65.8	—	0.72	0.80	0.08	5.2
	next	46.4	58.7	98.0	98.0	—	0.47	1.00	0.53	51.6
	previous	0.3	0.4	53.3	59.7	0.2	0.00	0.89	0.89	53.0
64, seed 2	current	54.7	47.5	55.9	67.4	—	0.81	0.83	0.02	1.3
	next	45.2	60.0	98.1	98.2	—	0.46	1.00	0.54	52.9
	previous	0.0	0.4	51.8	59.9	0.1	0.00	0.86	0.86	51.8
64, seed 3	current	48.2	46.6	51.7	68.2	—	0.71	0.76	0.05	3.5
	next	59.0	61.8	98.7	98.8	—	0.60	1.00	0.40	39.7
	previous	0.3	0.5	49.0	60.9	0.2	0.00	0.80	0.80	48.7
128, seed 1	current	8.4	6.4	8.4	16.5	—	0.51	0.51	0.00	0.1
	next	34.1	33.1	86.3	88.0	—	0.39	0.98	0.59	52.2
	previous	0.0	0.1	12.2	19.6	0.0	0.00	0.62	0.62	12.2
128, seed 2	current	9.1	6.9	9.3	16.8	—	0.54	0.55	0.01	0.1
	next	36.6	34.5	85.2	86.9	—	0.42	0.98	0.56	48.6
	previous	0.0	0.2	12.8	18.0	0.0	0.00	0.71	0.71	12.8
128, seed 3	current	8.4	7.0	8.4	16.3	—	0.52	0.52	0.00	0.0
	next	40.1	32.9	85.6	87.1	—	0.46	0.98	0.52	45.5
	previous	0.1	0.6	15.9	22.0	0.1	0.01	0.72	0.72	15.7

goes, but not how little it steers, and the network keeping less of its own row than the UFM predicts (Section 3.3) contributes to the difference.

Table 9: **Margins and steering, from saved activations.** Per target and width: the own-row cosine c of the DiM direction d; the median normalised margin $u = |z|/(s\,\|W_j\|)$ of the clear outputs; the relative flip rate of d (its flip rate divided by the readout row's) at $\alpha = 2$, in and out of distribution, and of a norm-matched random direction in distribution; the relative flip rate of d that the UFM's own-row cosine predicts from each model's margins, at $\alpha = 2$ and $\alpha = 4$; the percentage of clear bits flipped by d and by a random direction at $\alpha = 4$; and the percentage of holding(x) outputs flipped when steering along d for clear(x) at $\alpha = 2$ and $\alpha = 4$. Medians over 3 seeds. ID: in distribution; OOD: out of distribution.

			relative flip rate, $\alpha = 2$			UFM-predicted, d		d / random (%)	holding(x) along d (%)	
Width, target	c	u median	d, ID	d, OOD	random, ID	$\alpha = 2$	$\alpha = 4$	$\alpha = 4$, ID	$\alpha = 2$	$\alpha = 4$
64, current	0.85	1.74	0.71	0.89	0.002	0.89	1.00	98.8 / 0.55	0.0	0.0
64, next	0.24	0.38	0.61	0.49	0.11	0.77	0.93	79.5 / 20.1	11.6	30.9
64, previous	0.17	1.82	0.007	0.18	0.003	0.054	0.34	2.5 / 0.63	1.3	32.9
128, current	0.86	2.54	0.41	0.87	0.000	0.68	0.98	93.5 / 0.01	0.0	0.0
128, next	0.24	0.75	0.38	0.34	0.067	0.60	0.80	60.0 / 11.4	8.6	14.9
128, previous	0.28	2.60	0.010	0.27	0.001	0.018	0.065	1.6 / 0.06	0.0	6.5

Steering one block earlier. Section 4.4 steers at the output of the first block, position 28, along the unit DiM direction of clear(x) computed there on the fitting split, with the norm set by that site's activation scale. It reads the readout after the second block on the test-split examples where clear$_t(x)$ is false. For each model the statistic is the change in the clear(x) and holding(x) logits, averaged over the five x, and its z score against 50 norm-matched random directions applied to the same examples. We fixed three criteria before the test split was read, after an exploratory look at the fitting

Table 10: **Steering at matched realised L_2 norms, state-tracking sweep.** Flip rate (%) per direction at $\alpha = 2$, clear-state predicate, 27 models, in-distribution site, label-free baselines included. The readout row is algebraically optimal for its own logit at fixed L_2 norm: the maximum, not a competing method. Language model: Table 13, scoped in Section 7.

Direction	Flip rate (%)
State probe w_{state}	39.4
Output probe w_{output}	72.1
Next-state probe w_{next} (timing control)	51.1
Readout row W_j	74.3
DiM d	48.8
$\rho(w_{\text{output}}) - \rho(w_{\text{state}})$ (pp)	32.7
$\rho(w_{\text{output}})/\rho(w_{\text{state}})$	1.83×

and selection splits: $z_{\text{holding}} > 3$ in every shifted-target model (met, 12 of 12), $z_{\text{clear}} > 3$ in every current-state model (met, 6 of 6), and $|z_{\text{clear}}| < 2$ in every previous-state model (not met, 4 of 6). The previous-state z_{clear} is -0.2 to 0.4 at width 64 and 0.5 to 2.2 at width 128. Strengths $\alpha = 1$ and $\alpha = 4$, each against 20 random directions, give the same pattern. An independent re-implementation reproduced every audited value.

Random-direction controls. Tested on *any* change of the output move argmax, a random direction changes the Othello move more often than the occupancy state probe w_{state}, 9.57% against 5.57%. The margin rule of Section 4.1 accounts for this too ($r = 0.973$ on random directions). A random direction has a component on all 65 move rows, whereas the probe carries one square's geometry. The stricter event of setting a complete 36-bit Blocksworld state, with every untouched fact preserved, is out of reach for random perturbations. Of $16{,}384$ same-norm random perturbations, 0 reach the target, whereas projected gradient descent at the same norm reaches it on 100 of 256 cases. We measure this event at the upstream site of Appendix J.3, which has a nonlinearity before the readout.

F Without the counting shortcut

The counting shortcut holds exactly on the standard encoding, and the rotation survives an encoding that removes it, at width 64 in three seeds (Section 5.1). Every Blocksworld action names the blocks it touches, so the counting shortcut of Section 2 holds exactly. We checked this by enumerating all 866 reachable states and $2{,}090$ transitions of the standard encoding. The shortcut is an identity of rank 25, and a constant-depth shortcut in the sense of Liu et al. (2023). An encoding that hides unstacking's source block removes it. An earlier, exploratory run on this encoding is in Appendix J.1.

The source-hidden replication. Table 11 collects the source-hidden replication of Section 5.1. Its models reach 99.28–99.79% complete-state accuracy under the two deterministic targets. Its current-target models score 0.99–1.00 count-ambiguous pair accuracy, against 1.00 for the current-target standard-encoding controls trained on the same episodes. Against those controls, the hidden probes sit at the control's level under the next-state target, with paired $|\Delta| \leq 0.06$ on both rows in every seed. Under the previous-state target they keep 0.04–0.17 more of their own row. The paired criterion is therefore confirmatory for the next-state models, and we do not rely on it for the previous-state ones.

Accuracy thresholds. We hold the next-state models to an excess-loss threshold, an upper bound on their mean per-bit cross-entropy above the Bayes-optimal value under random legal play. The standard-encoding control must stay under a fixed bound, and the source-hidden next-state model within a fixed allowance of its same-seed control. Table 15 records when we fixed this threshold and what that means for the first seed. At nearly equal accuracy, the source-hidden next-state model sits further above the Bayes-optimal cross-entropy than its control, so it meets the allowance but not the fixed bound. The width-128 models, which show the same pattern, are exploratory.

Table 11: **Source-hidden replication at width 64** (Section 5.1), 3 seeds per target, minimum–maximum over seeds. Cosines of the DiM direction d for the current-state fact clear(x) with its own row and with the coupled row holding(x); the cosine criterion holds in 6 of 6 shifted-target comparisons. Relative flip rates (flip rate over the row's) for d, the state probe w_{state} and the output probe w_{output} are for the previous-state models, in distribution at $\alpha = 2$. Complete-state accuracy pools the current- and previous-state models. Count-ambiguous pair accuracy is measured on 256 episode pairs that no function of the action multiset separates, for the current-target models; excess cross-entropy is for the next-state models, in nats per bit above the Bayes-optimal value. Paired differences are against the standard-encoding control of the same seed.

	current	next	previous
Own-row cosine, d	0.883–0.894	0.295–0.308	0.286–0.376
Coupled-row cosine, d	$[-0.008, 0.001]$	0.666–0.677	0.555–0.609
Relative flip rate, d	—	—	0.05–0.09
Relative flip rate, w_{state}	—	—	0.014–0.022
Relative flip rate, w_{output}	—	—	0.87–0.93
Complete-state accuracy (%)	99.28–99.79 (current and previous targets)		
Count-ambiguous pair accuracy	0.99–1.00 (control 1.00)	—	—
Excess cross-entropy (nats/bit)	—	0.028–0.032 (control 0.017–0.018)	—
Paired difference in shortfall from Bayes-optimal (pp), max	—	0.6	—
Paired $\lvert\Delta\rvert$ in own or coupled cosine, max	—	0.06	—
Paired own-row excess over the control	—	—	0.04–0.17

G Othello: centring, task performance and controls

The mean-centred sign is a well-defined geometric quantity even though the next-move readout predicts poorly. A softmax readout is invariant to adding a common vector to all its rows (Park et al., 2024; 2025). Such a shift can carry a raw row cosine across zero while leaving every output probability unchanged up to floating-point error, so we mean-centre the next-move rows in Figure 3. The occupancy readout is a per-bit sigmoid readout on which no such shift acts, and there centring is the identity. After 20 moves, two corner squares of the next-move board are near zero.

Task performance. The occupancy readout, trained on the occupancy after the next move, reaches 98% bit accuracy, the level of a persistence baseline at 98%. Under random-legal play, persistence is Bayes-optimal rather than merely a baseline, so this readout cannot improve on it. The next-move readout's top-1 move is legal in 77.5% of positions, and it puts 57.7% of its probability on legal moves. Its top-1 accuracy is 7.1%, against a Bayes-optimal 9.4% under random-legal play, which chooses uniformly among legal moves. The next-move readout is thus a weak predictor. The sign, however, concerns only the geometry of d and a row of fixed weights, which is defined at any accuracy, so for these two readouts we report geometry only and make no steering claim.

Controls. The two Othello models are separately trained networks rather than weight-matched pairs, so the control is the fixed task and corpus. The margin-rule check of Figure 2b uses a different family of Othello models, with next-move readouts over a grid of three widths, three depths and two seeds, each probed on the same squares. We exclude a (model, square) cell when its state or output probe cannot be fitted, or fits its own training data with accuracy below 60%. This excludes 96 of 288 cells, all from the wider and deeper configurations.

H Do the labels explain the forward excess?

The labels account for where the forward targets send d, but not for how strongly, relative to the other rows, they align it with the holding(x) row, and the backward control that would isolate this excess

failed. This appendix gives the design and every model's values behind the Limitations paragraph (Section 7).

Design. In random legal play, the hand alternates with position, so a holding fact is a function of parity, and an excess could be read off position. We therefore retrain under a policy with an always-legal no-op, which breaks the alternation and makes up 23% of training tokens. We also add targets two actions ahead (Y_{t+2}) and two back (Y_{t-2}). The forward targets are then Y_{t+1} and Y_{t+2}, and the backward ones Y_{t-1} and Y_{t-2}. For each model we take the normalised alignment with the holding(x) row, the holding(x) entry of the vector of d's cosines with all 36 readout rows, normalised to unit length.

We compare this alignment with the range of label-only predictions, taken over 37 fits to the same labels. These are 36 UFM fits, over 4 weight decays, 3 feature dimensions and 3 seeds, and an identity map whose features are the smoothed label logits. A model exceeds the labels when its alignment lies above that range by at least 0.06, twice the noise floor of a control under random legal play. Table 12 gives every model's alignment, range and margin.

Forward targets. Under the no-op policy the UFM puts this alignment at 0.14–0.20 for every target other than Y_t. Both forward targets reach at least 0.46 and exceed the range by more than 0.06 in all 12 models (Figure 4). The width means of the excess differ by at most 0.03, so it barely depends on width, and the current-state control stays within 0.06 of its range.

Backward targets. Backward targets have label statistics that match the forward ones, yet they behave differently. No two-step backward model exceeds the range (0 of 6). Its whole row vector agrees with the UFM more closely than the two-step forward model's does, and under random legal play it sits below the range. One-step backward models exceed the range by more than 0.06 only at width 64. At width 128 they exceed it by 0.02–0.05, around the control's noise floor, so this backward excess shrinks with width, as the own-row shortfall of Section 3.3 does. The forward excess does not shrink.

We pre-specified that the backward targets would stay with the labels. In each backward target, at least five of its six models would lie within the control's noise floor of the range, which is half the margin, and no backward model would exceed the range by 0.06. They did not. The one-step models at width 64 exceed the range by more than 0.06, and those at width 128 by up to 0.05. Two-step models fall below it, all three at width 128 and one at width 64 by more than half the margin. The forward result therefore rests on the excess comparison alone, which scores forward and backward targets alike.

Two readings of the excess. One reading is look-ahead, a future step represented before it is emitted, as reported for planning language models and a chess network (Men et al., 2024; Jenner et al., 2024). For us this is an analogy only, since our future state is the target itself rather than a later output. The other is that the forward holding label depends on $\text{clear}_t(x)$ and an empty hand, so the readout can compute it from the present-state direction. No criterion scored either reading, and both remain compatible with our data.

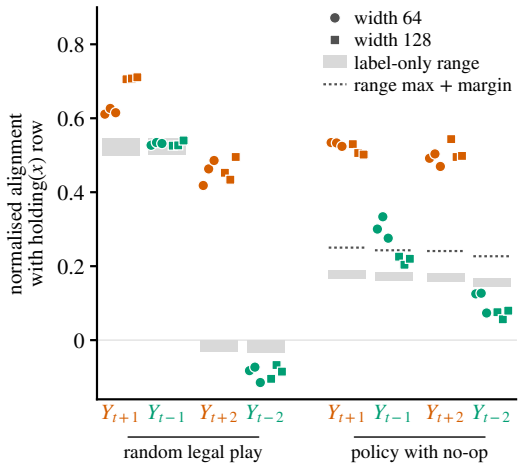

Figure 4: **Forward targets exceed the label-only range by the margin; among backward targets, only the one-step models at width 64 do so once a no-op breaks the hand's alternation.** Each mark: one model's normalised alignment with the holding(x) row, the holding(x) entry of the unit-normalised vector of d's cosines with all 36 readout rows. Grey bands: the range of label-only predictions for each target, over UFM fits and an identity map on the label logits (Table 12). Left, random legal play; right, a policy with an always-legal no-op, which breaks the hand's alternation with position. Dotted: the range's maximum plus 0.06.

Table 12: **Normalised alignment of d with the holding(x) row against the range of label-only predictions, by label policy, target and width.** The range is taken over 37 label-only comparators fitted to the same labels (36 UFM fits over 4 weight decays, 3 feature dimensions and 3 seeds, and an identity comparator on the smoothed label logits), one range per target shared by both widths. Margin: the alignment minus the range's upper edge. Above margin: models whose margin reaches 0.06, twice the noise floor of a control under random legal play. Agreement: cosine between the model's and the UFM's 36-coordinate row vectors, median over comparator seeds. Entries are [minimum, maximum] over the 3 seeds of a width, one value where they coincide. The no-op controls sit at most 0.030 outside the range, against their own noise floor of 0.034; the width means of the forward margins differ by at most 0.024.

Target	Width	Alignment	Label-only range	Margin	Above margin	Agreement
Policy with a no-op						
current (control)	64	[−0.23, −0.18]	[−0.20, −0.14]	[−0.08, −0.04]	0/3	[0.97, 0.98]
	128	[−0.23, −0.19]	[−0.20, −0.14]	[−0.08, −0.04]	0/3	[0.98, 0.99]
next	64	[0.52, 0.53]	[0.17, 0.19]	[0.33, 0.34]	3/3	[0.75, 0.76]
	128	[0.50, 0.53]	[0.17, 0.19]	[0.31, 0.34]	3/3	[0.82, 0.83]
Y_{t+2}	64	[0.47, 0.50]	[0.16, 0.18]	[0.29, 0.32]	3/3	0.65
	128	[0.50, 0.54]	[0.16, 0.18]	[0.31, 0.36]	3/3	[0.70, 0.73]
previous	64	[0.28, 0.33]	[0.16, 0.18]	[0.09, 0.15]	3/3	[0.85, 0.91]
	128	[0.20, 0.23]	[0.16, 0.18]	[0.02, 0.05]	0/3	[0.96, 0.97]
Y_{t-2}	64	[0.07, 0.13]	[0.15, 0.17]	[−0.09, −0.04]	0/3	[0.93, 0.95]
	128	[0.06, 0.08]	[0.15, 0.17]	[−0.11, −0.09]	0/3	0.96
Random legal play						
next	64	[0.61, 0.63]	[0.50, 0.55]	[0.07, 0.08]	3/3	—
	128	0.71	[0.50, 0.55]	[0.16, 0.17]	3/3	—
Y_{t+2}	64	[0.42, 0.49]	[−0.03, 0.00]	[0.42, 0.49]	3/3	[0.49, 0.55]
	128	[0.43, 0.50]	[−0.03, 0.00]	[0.43, 0.50]	3/3	[0.61, 0.63]
previous	64	0.53	[0.50, 0.55]	[−0.02, −0.01]	0/3	—
	128	[0.53, 0.54]	[0.50, 0.55]	[−0.02, −0.01]	0/3	—
Y_{t-2}	64	[−0.11, −0.07]	[−0.03, 0.00]	[−0.11, −0.07]	0/3	[0.95, 0.96]
	128	[−0.10, −0.07]	[−0.03, 0.00]	[−0.10, −0.07]	0/3	0.97

I A FINE-TUNED LANGUAGE MODEL

On a language model whose readout sits behind a nonlinearity, the output probe's advantage over the state probe survives on one checkpoint, label-free directions steer better, and a matched pair leaves its pre-specified steering criterion unsupported. The checkpoint is a fine-tuned base Qwen2.5-0.5B, read at its validation-selected site and strength (Table 13). At matched realised norms the output probe flips more answers than the state probe. Two label-free directions, the unembedding contrast and the sample-local gradient of the pair margin, flip more than the output probe.

Table 13: **Steering a fine-tuned language model at matched realised L_2 norms.** Flip rate (%) at the answer slot of a LoRA fine-tune of the base Qwen2.5-0.5B (Qwen Team, 2024) on this task, read at decoder layer $L22$ with $\alpha = 4.0$ over 384 answer slots; layer and steering strength were selected on a validation split and norms are matched on the realised per-dimension RMS. The site has a nonlinearity between it and the unembedding, so equation 1 does not govern it. Both label-free directions beat the output probe, and steering along w_{output} moves pair accuracy from 80% to 73%.

Direction	Flip rate (%)
State probe w_{state}	13.54
Output probe w_{output}	32.29
DiM d	28.12
Unembedding contrast (label-free)	44.01
Sample-local pair gradient (label-free)	65.36
$\rho(w_{\text{output}})/\rho(w_{\text{state}})$	2.38×
Correction / corruption (w_{output})	12.76 / 19.53

A matched pair. We then trained matched pairs of LoRA fine-tunes of the same base model (Qwen Team, 2024), whose affine readouts predict the current or the next state (Table 14). Its criterion concerns steering rather than the rotation. It uses the DiM direction's contrast-normalised flip rate $\rho/\rho_{\text{contrast}}$, its flip rate divided by that of a label-free unembedding contrast. We pre-specified that this rate would be lower under the next-state target in all 3 pairs. It was lower in 2 of 3, so the criterion is unsupported. In the pair that misses it, the next-state model fell below its accuracy threshold, and one of the pairs that meet it rests on a near-floor denominator. The seeds differ only in the LoRA initialisation, with identical training data and minibatch order. We scored every pair in single precision (Appendix K).

Beyond the criterion, the output probe's contrast-normalised flip rate is the largest of the three probe directions in 6 of 6 models, and the label-free unembedding contrast exceeds it in 4 of 6. The two exceptions are the models of the near-floor pair.

Table 14: **Steering a matched pair of LoRA fine-tunes of the base Qwen2.5-0.5B.** Per pair, at its validation-selected decoder layer and steering strength α (the requested L_2 norm is αs): the flip rate ρ of each direction over 383 answer slots per model, and $\rho/\rho_{\text{contrast}}$, the flip rate divided by that of the label-free unembedding contrast (*contr.*), a baseline rather than a maximum; *grad.* is the sample-local gradient of the pair margin. Primary criterion: a lower DiM $\rho/\rho_{\text{contrast}}$ under the next-state target, required in all 3 pairs and met in 2. Secondary criterion: a larger $[\rho(w_{\text{output}}) - \rho(w_{\text{state}})]/\rho_{\text{contrast}}$ under the next-state target. Last column: the next-state model's shortfall from Bayes-optimal accuracy, in single precision. Pair 2's current-state contrast flip rate sits just above the floor (0.05), so its ratios exceed one.

		Flip rate ρ					$\rho/\rho_{\text{contrast}}$					
Pair (site)	Target	w_{state}	w_{output}	DiM d	contr.	grad.	DiM d	w_{state}	w_{output}	primary	secondary	shortfall from Bayes-opt. (pp)
1 (L22, α=8)	current	0.232	0.394	0.355	0.598	0.744	0.59	0.39	0.66	fail	pass	11.53[†]
	next	0.243	0.522	0.473	0.663	0.802	0.71	0.37	0.79			
2 (L18, α=8)	current	0.175	0.178	0.141	0.063	0.739	2.25	2.79	2.83	pass	pass	7.94
	next	0.399	0.567	0.446	0.428	0.958	1.04	0.93	1.32			
3 (L22, α=8)	current	0.360	0.561	0.462	0.642	0.789	0.72	0.56	0.87	pass	fail	9.42
	next	0.342	0.413	0.376	0.577	0.802	0.65	0.59	0.71			

[†] above the pre-specified 10 pp accuracy threshold; the pair is kept and scored under both criteria.

J Additional experiments

Three further experiments probe the scope of our claims: an earlier run on the source-hidden encoding, an auxiliary loss that imposes row alignment, and state editing at an upstream site.

J.1 An earlier source-hidden run (exploratory)

An earlier, independently trained run points the same way (Figure 5), but its source-hidden models fell short of their accuracy threshold as a group, so every value here is exploratory. The run compared source-hidden and standard-encoding models on the same episodes, under the current- and next-state targets, at 60 and 90 epochs. Its threshold requires a next-state model's shortfall from Bayes-optimal accuracy to stay close to that of its same-seed standard-encoding control. We read its checkpoints on development data, fitting directions on the fitting split and selecting the logistic constant and measuring flip rates on the validation split. Its standard-encoding models supply the independent control of Table 6, which bears on the target's effect rather than on the shortcut. Individually, 2 of the 6 source-hidden pairs clear the threshold, and they show a smaller output–state gap on average than the rest. The read is consistent with count-derived next-state prediction being easier at a matched norm than prediction on the source-hidden encoding.

A gap under the current-state target. The matched pairs isolate the target's contribution partly because the standard encoding's current-target models are essentially perfect. The two probes are then fitted to almost the same labels, the definitional case of Appendix A (the output bits are linearly separable, Appendix E), and their gap is near zero at 1.69 points. A source-hidden encoding breaks that coincidence. Models trained there on a *current*-state target, below the accuracy threshold, already show an output–state flip-rate gap of 16 points, with the state probe flipping 24.0–46.4% of bits against 46.5–56.4% for the output probe. In the starkest of them, a probe reading 98.6–99.0% on three predicates sits 62–65° off the row it should drive, a target-free instance of the decodability–steerability gap that observational studies report (Galeone et al., 2026; Billa, 2026). In this exploratory run, a target-independent misalignment and the target-dependent rotation coexist, and the output probe recovers the readout-aligned direction in both.

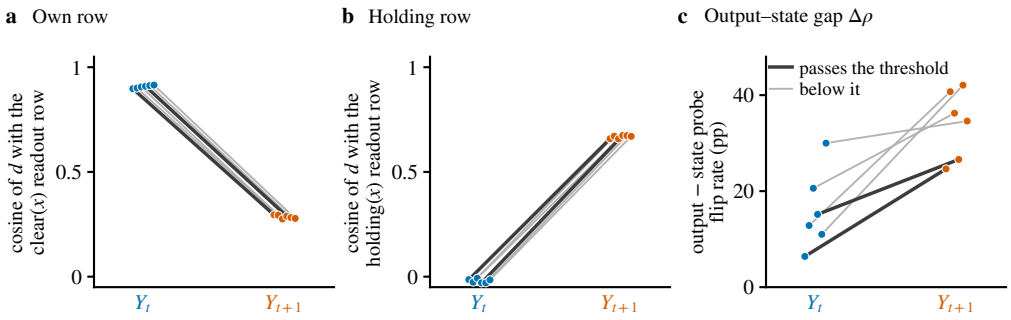

Figure 5: **An earlier source-hidden run shows the same rotation and a growing output–state gap.** Exploratory: development data, and models below their accuracy threshold as a group; the replication is in Section 5.1. One line per matched pair, from the current-state model (blue) to the next-state model (orange); dark: the 2 of 6 pairs that individually pass the accuracy threshold. **(a)–(b)** Cosines of d, the difference-in-means (DiM) direction for the current-state fact clear(x), with the clear(x) and holding(x) readout rows (the two axes of Figure 1b). **(c)** The output–state gap $\Delta\rho = \rho(w_{\text{output}}) - \rho(w_{\text{state}})$, in percentage points at $\alpha = 2$.

J.2 An auxiliary row-alignment loss

Row alignment can be imposed by an auxiliary loss early or late in training, and kept on at no measurable cost, but it decays toward the control once the loss is removed. Since a direction steers an output bit through its component on that bit's row (Section 4), we tested this on depth-2 next-state-target models trained like the sweep of Appendix A for 100 epochs, at two widths and two seeds (4 width/seed combinations). The loss, of weight 0.01, rewards alignment between a fact's probe direction and its own readout row. It acts only at in-distribution positions and is removed after a fixed window, and we read alignment at positions $40/41$, beyond the training length (Figure 6).

Row alignment peaks at 0.91 at the end of an early window, and at 0.80 at the first checkpoint after a late one, where the one checkpoint inside the late window reads 0.47. The control trained without the loss ends at 0.26. Models trained with the loss always on differ from the control by $+0.02$ points of held-out accuracy. After removal, row alignment in both windowed runs decays toward the control trajectory (0.40 and 0.58), while the always-on models keep 0.95.

Motivated by critical periods for input deficits (Achille et al., 2019) and for regularisation (Golatkar et al., 2019), we pre-specified the hypothesis that row alignment can be shaped only early in training. The data reject it, with 1 of 5 criteria met, since the late window also induces alignment well above the control. Decodability supervision also loses most of its gains within a few hundred steps of fine-tuning once its auxiliary heads are removed (Dingeto, 2026), so retention after removal must be measured separately from alignment while the loss is on.

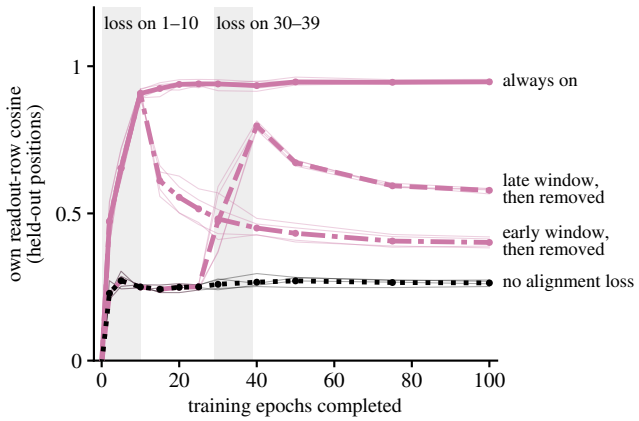

Figure 6: **An auxiliary row-alignment loss raises the own readout-row cosine in either window, but the gain decays after removal, and at epoch** 100 **the early window retains less than the late one.** The loss ($\lambda = 0.01$) pulls d, the difference-in-means (DiM) direction for clear(x), toward its own readout row. The early- and late-window conditions apply it for ten epochs (shaded) and then remove it; it stays on in a third condition and is absent in the control. A run's value is the median over the clear(x) facts of the cosine at held-out positions $40/41$; thick lines average the 4 (width, seed) runs, thin lines show individual runs. The two window conditions peak at 0.91 (early, at the end of its window) and 0.80 (late, read at the first checkpoint after removal, one epoch later); at epoch 100 they retain 0.40 and 0.58, above the control's 0.26.

J.3 State editing at an upstream site

At an upstream site, setting the state that the readout reports at one position does not carry over to the states the model computes next. We edit current-state-target state-tracking models with two blocks, at the residual stream after the first block. A full transformer block, with its nonlinearities, thus lies between the edit and the readout, and equation 1 does not govern this site.

Complete counterfactual current states are reachable by optimisation here. In an exploratory pilot at the steering strength selected on the fitting split ($\alpha = 16$), norm-constrained projected gradient descent (PGD; Madry et al., 2018) within the L_2 ball induces the complete counterfactual 36-bit state on $64/64$ validation pairs in each of 2 checkpoints. Every untouched fact is preserved. The edit does not propagate, however. In the propagation test the same edit reaches the complete state at the edited position in all 6 models, and at neither of the 2 successor positions in any of them. Even an edit that makes the current-position logits identical to those of the counterfactual world yields a different future. This is consistent with the next state being computed from the activation rather than from the logits.

Illegal target states do not explain the failure to propagate. The edit targets are states replayed from the simulator, so they are legal worlds by construction. Legality held in $64/64$ of the edited cases, and the unedited factual state is legal too. We use this site only for the result reported here and for the random-direction control of Appendix E.

K Pre-specified criteria and outcomes

Every figure, table and number in this paper is generated from the released result files by scripts in the supplementary material, whose README maps each figure and table to its script and input files. Table 15 lists each pre-specified criterion in words, its outcome and what it means, followed by the post hoc and exploratory analyses; its caption gives when the criteria were fixed. A criterion is *pre-specified* when we fixed it and its threshold before the evaluation it scores. A *post hoc* analysis was specified after the models had been trained, and it describes rather than tests. An *exploratory* one uses development data or models below their accuracy threshold.

Two further facts bear on how the results can be combined. Development results are not independent of one another: they draw on a pool of 36 development states, of which an eight-state subset underlies at least eight analyses, and two checkpoint pairs underlie more than ten. In the language-model pairs, half precision missed its numerical tolerance before any criterion value was seen, with next-state accuracy shortfalls of 12.03 and 7.89 points in two pairs (the third was not run in half precision), so we scored every pair in single precision with the tolerance, strength grid, estimators and criteria unchanged.

Table 15: **Pre-specified criteria and what they showed.** Each criterion was fixed before the data it scores were seen, except the accuracy threshold of the source-hidden replication, fixed after the first seed's task performance was seen and before any geometry, flip rate or criterion value of the other seeds, and the six criteria of Section 4.4 and Appendix D, fixed after an exploratory look at the fitting and selection splits and before the test split was read. Post hoc and exploratory rows are descriptive.

Prediction	Outcome	What it means
Under the previous-state target, d's own-row cosine drops, its coupled-row cosine rises, and the UFM predicts the coupled-row cosine (Section 3.3)	supported, and reproduced by an independent re-implementation	an uncertain target is not needed
Own-row drop with coupled-row rise for both shifted targets over six seeds (Appendix C)	holds in 6 of 6 seeds, on the development split	the rotation holds beyond three seeds
Under the no-op policy, forward targets exceed the label-only range by the margin 0.06 (Appendix H)	met in 12 of 12 models	the labels do not account for the forward strength
Under the no-op policy, in each backward target, at least five of six models lie within half the margin of the label-only range and none above it by the margin (Appendix H)	not met: one-step models at width 64 exceed the range by the margin, and two-step models fall below it	the forward claim rests on the excess comparison alone, which scores forward and backward targets alike
The state probe reads through a subspace the readout ignores (Table 5)	rejected	uninformative, since the threshold lay below the accuracy of reading clear_t from Y_{t-1}
Some auxiliary-target design makes the two steering-gain predictors of Appendix B disagree	0 of 3 required designs found	no searched design tells the two predictors apart
Under both shifted targets, the source-hidden models at width 64 align d more with the coupled row than with their own (Section 5.1); accuracy threshold fixed late, see caption	met in 6 of 6 comparisons	the first seed does not test the threshold; the other two seeds were each evaluated once against it
In the language-model pairs, the DiM direction's contrast-normalised flip rate is lower under the next-state target in all 3 pairs (Appendix I)	lower in 2 of 3 pairs	unsupported
Under a reparametrisation-invariant cosine, d is closer to the holding(x) row than to its own in every shifted-target model, with a positive own-row value, and the reverse under the current target (Appendix D)	met in 12 of 12 and 6 of 6 models	the alignment is not a coordinate artefact
Steered one block earlier, the clear(x) direction raises the coupled holding(x) logit ($z > 3$) in every shifted-target model and its own output ($z > 3$) in every current-target model (Section 4.4)	met in 12 of 12 and 6 of 6 models	later computation reads the direction; under shifted targets it reaches the coupled output
… and leaves the previous-state clear(x) output within the random band ($\lvert z\rvert < 2$)	not met: 4 of 6 models	the scored output moves slightly at width 128, in almost no answers
Row alignment can be shaped only early in training (Appendix J.2)	rejected (1 of 5 criteria met)	alignment can be imposed late too, and decays once the loss is removed
Post hoc and exploratory analyses		
One-step targets under random legal play (Table 12; post hoc)	—	descriptive
Margins, flip rates at every strength, random directions, coupled-output flips, reading clear_t from Y_{t-1}, row-space energy, self-duality, the UFM cosine with margins (Table 9; post hoc, from saved activations)	—	descriptive, as is every quantity citing them
Which object moves, for the clear(x) and holding(x) rows (Appendix D; post hoc)	the attribution reverses in an activation-aligned frame	descriptive
Steering-gain predictor comparison on six models (Appendix B; exploratory)	—	descriptive
Source-hidden models at width 128 (Appendix F; exploratory)	excluded before testing, because their standard-encoding control exceeded the fixed bound	same pattern
Earlier source-hidden run (Appendix J.1; exploratory, development data)	below its accuracy threshold	points the same way
Upstream editing with PGD on 2 checkpoints (Appendix J.3; exploratory pilot)	—	descriptive