Ex 2.1.6: anchoring *red* in a transformer

> Our first anchored transformer, with a single attractive term. The anchor moved the activations to the chosen direction, and cost the task nothing measurable. But it moved the whole color cube, rather than *red* in particular.

This is the first anchored transformer. During training we add one regularizer term that pulls the residual stream activations of *red-labeled* equations toward a fixed direction $\hat{v}_{\text{red}}$. Then we ask two questions: did the concept land where we put it, and what did that cost the task?

▶ **Notation…**

As the first anchored experiment, it opens D2.1 proper. M1 established anchoring in autoencoders, and everything since M2/Ex-2.1.1 has been un-anchored groundwork: finding a testbed where the task is solved and the concept geometry is legible, so that the effects of an anchor can be attributed to it.

The testbed is the word-level `v216` condition from Ex-2.1.3: 216 grid colors, **one token** each, `name + name = name` equations, d64-L4. Held-out baseline accuracy is ≈ 1.0 and the color cube is linearly decodable from the embeddings, so there is headroom to lose and a known un-anchored geometry to compare against.

The labels are sparse, noisy, and binary, following M1/Ex-1.7 via M1/Ex-2.9.1. Each line is labeled *red* with probability `redness(op1)`^{8} `× 0.08`. Only strongly red first operands are ever labeled, and even they usually aren't.

Samples were atomic in the autoencoder experiments, but now that each sample is a sequence, we need to decide *what* to label. Here, labels attach to equations, and the anchor term pulls every position of a labeled equation's prompt equally, at every layer. That matches the shape of natural language, where a span of text is labeled rather than a token. It also leaves a question for the training dynamics to answer: the pull is undifferentiated but the task loss resists it unevenly, so does the concept condense at the position that computes it, or smear into a broadcast "this line mentions red" feature?

The schedule comes from issue #10 and M1/Ex-2.9.3: regularizer protection has to outlive the heat of the optimizer. So the *anchor* weight holds at peak until the LR decay is essentially done, then anneals to a floor above zero.

The architecture helps too. Our simplified nGPT keeps every residual-stream state on the unit hypersphere, so the cosine-geometry anchor term from M1 transfers unmodified. In an unnormalized residual stream a cosine term constrains direction and leaves magnitude free, so it would need a companion term with its own weight to keep activation scale in hand — one more coefficient to tune. Norms pinned at 1 remove that question.

We leave out the *anti-anchor*, *anti-subspace*, and *separate* terms, and any intervention (suppression, ablation, or redirect): one anchor term only. Nothing reserves the axis, so this experiment asks whether it *happens* to carry red alone. The repulsive terms matter once we intervene, when a stray concept sharing the axis would show up as a side-effect.

> **How to read this report**
>
> This was a preregistered skeleton. The method, hypotheses, and decision thresholds below were written and frozen before the experiment ran, and the results replaced the placeholders in place, so reading it is a prediction → observation diff. Anything conceived after seeing the data is under *Exploratory analyses*, marked post hoc.
>
> Two amendments were made after the freeze and before the run, neither touching a threshold. The noise floors quoted under H2(b) and H4 were corrected — the first draft divided the per-cosine spread by the 27 probe lines of a color as well as the 5 layers, which the measurement cannot do; the passage under H2(b) says what the correction changes. And the mean alignment over colors was added beside the margin as a recorded diagnostic, on the reasoning that a margin is a contrast and so cannot distinguish "nothing moved" from "everything moved". It carries no gate, but it turned out to be what the results are about.

Method

Task and corpus

Identical to the `v216` condition of ex-2.1.3: the six-level sub-grid `(0, 3, 6, 9, 12, 15)` of the 16-level cube (216 colors, one opaque token each), equations `name + name = name` with exact integer mixing, 100 k lines, 20% of distinct closed pairs held out. The eval sets are the same three: `named_seen`, `named_holdout`, and `open` (pairs whose mix has no name).

We add one thing: an alignment probe set. For each of the 216 colors, it builds one probe line per closed partner of that color, meaning the 27 pairs whose mix has a name, with the color as the first operand. Closed pairs guarantee every probe line has a named answer, so the measured answer position corresponds to a real token. Using all 27 makes the probe set exhaustive: no sampling, no seed, and every run measured on identical lines, drawn from seen and held-out pairs alike. Per-color alignment statistics average over op2 rather than conditioning on it.

At op1 that average is over identical values, and deliberately so: a causal model's first position attends to itself alone, so a color's op1 alignment is the same on all 27 of its lines. The statistic H2 and H4 score is therefore a property of the token, measured without sampling error from the partner — but also without any averaging to quiet the measurement, which is what sets the noise floor quoted under H2(b) and H4. Averaging over partners does its usual work at the later positions, which H3 reads.

Labels

A line is labeled *red* by a coin flip weighted by the redness of its first operand: `p = redness(op1)⁸ × 0.08`, where `redness = r·(1 - g/2 - b/2)`, one independent draw per line.[1] This is the `RED_PROB` of M1, applied to the color of the first operand only. A red op2 or a red answer never triggers a label, which keeps the ground truth unambiguous for the question of *where* the concept should condense. An either-slot labeling scheme is queued as a follow-up.

Labels are redrawn per visit, as in M1, so they differ from epoch to epoch, and the expected pull on a color is proportional to its label affinity. The graded response of H2 is a further claim on top of that. The affinity spans four orders of magnitude, so a response that stayed proportional to it would be measurable on only the reddest few dozen colors. The grading H2(b) predicts comes instead from generalization: colors partway red moving partway. M1 is the precedent. With labels drawn from this same affinity, the damage it measured under intervention graded as low powers of its HSV similarity-to-red — `sim³` for ablation, `sim²` for suppression — far flatter than `redness⁸`. The ceilings under the hypotheses make this quantitative.

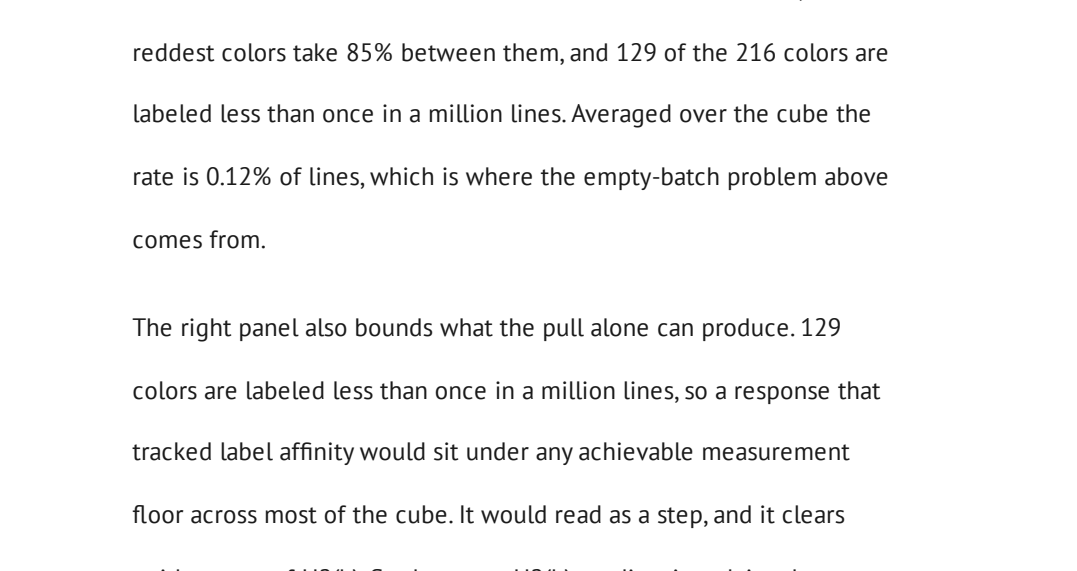
Batch composition

Samples are packed sequences, not single equations. An equation is 6 tokens at word level and a sequence is 64 tokens, so a batch of 64 sequences carries about 683 equations. The corpus-mean label rate is 0.12%, which works out to 0.81 labeled lines per batch, with 56% of batches carrying at least one. That is denser supervision than M1 ran with, where the term fired on roughly 6% of batches.

At that density we skip the label balancing that issue #10 recommends. Labels are plain independent draws, one per line. The anchor term is normalized by however many labels a batch happens to hold; that normalization is inherited from M1. Both the original implementation and our reproduction (`sca.colorcube.loss_terms`) take a label-weighted mean with a small ε in the denominator. So the 44% of batches holding no label contribute nothing rather than a 0/0. The gradient of the term is therefore noisy between batches, but that noise is inherent to sparse labels. If the H4 trajectories show the term failing to take hold, balancing is still available as a variance-reduction step.

Skipping it also keeps the comparison clean. Balanced batching over-represents red first operands, so the anchored conditions would train on a different corpus from the control, and every accuracy difference in H1 would carry an alternative explanation. As it stands, the control and the anchored conditions see identical batches in identical order for a given seed; the only difference between them is λ.

▶ **A more realistic labeling variant…**



The label distribution over the 216 colors. Left: the cube with mark area proportional to a color's share of all labeled lines, floored so that never-labeled colors still show. Right: per-color label probability against redness (log scale, the eighth-power curve behind it), with the dashed line at the corpus mean and the grey area giving the cumulative share of labels below each redness.

The eighth power is what makes this a *concept* label rather than a redness readout. Pure red alone takes 31% of all labeled lines, the ten reddest colors take 85% between them, and 129 of the 216 colors are labeled less than once in a million lines. Averaged over the cube the rate is 0.12% of lines, which is where the empty-batch problem above comes from.

The right panel also bounds what the pull alone can produce. 129 colors are labeled less than once in a million lines, so a response that tracked label affinity would sit under any achievable measurement floor across most of the cube. It would read as a step, and it clears neither gate of H2(b). So the curve H2(b) predicts is a claim about generalization: the anchor catching *red* rather than the labeled tokens, with colors partway red moving partway whether or not they were ever labeled. The model only ever sees Bernoulli draws from this distribution, and nothing stops it from treating "red enough to be labeled sometimes" as a threshold. That outcome is what (b) scores against.

> **The concentration is a risk we are choosing to accept**
>
> The weights behave like about 12 distinct colors rather than 216, and pure red alone takes 31% of the labels. So the anchor's evidence is a dozen or so first operands, repeated for 100 epochs. An axis could learn *this token* instead of *red* and still clear H2(a) and (c).
>
> We keep M1's exponent anyway. Ex-2.1.6 asks whether the M1 result transfers to a transformer, and moving the label distribution at the same time as the architecture would leave nothing to attribute a difference to. It is also the closer analogue of a document-level label in natural language, which fires on a small and unrepresentative slice of the corpus. H2(b), the grading test, separates a memorized exemplar from a generalized concept, which is a good part of why it is a gate rather than a diagnostic. A flatter exponent is queued in `todo-science.md` as the follow-up that would settle it.

Model and anchor term

The d64-L4 simplified nGPT as used from ex-2.1.1 onward, unchanged. The anchor direction is $\hat{v}_{\text{red}} = e_1$, the first basis vector of the 64-dimensional residual stream — the vector $(1, 0, 0, \ldots)$, matching M1. The residual operations of this architecture (LERP toward normalized sub-module output, scalar gains) are rotation-equivariant, so choosing a basis vector is a convenience rather than a hint to the model.

For labeled lines, the anchor term is

$$\mathcal{L}_{\text{anchor}} = \frac{1}{|\mathcal{P}|\,(L+1)} \sum_{t\in\mathcal{P}} \sum_{\ell=0}^{L} \left(1 - \cos(h_{\ell,t},\, \hat{v}_{\text{red}})\right)$$

where $h_{\ell,t}$ is the residual-stream state at layer ℓ, position t, averaged over all $L+1 = 5$ layers (the token embedding plus the four block outputs) and over $\mathcal{P}$, the positions of the labeled equation's **prompt**: `op1`, `+`, `op2`, `=`. Unlabeled lines contribute nothing, and no other term is added.

The answer token and the newline are inside the labeled equation but outside $\mathcal{P}$. We exclude them because of a confound: the answer of a red-labeled line is itself reddish, since a red op1 drags the mix. An anchored answer position would therefore align for two reasons at once, and we could not tell them apart. The exclusion costs nothing that we are measuring: the four pulled positions still receive an identical, undifferentiated pull, so the condensation question of H3 stands, and the answer position becomes a clean read of how far alignment spreads on its own.

What the exclusion does not do is spare the state that emits the answer. The model is causal, so the answer token is predicted from the state at `=`, a position inside $\mathcal{P}$ that is pulled at every layer, including the final one the logits are read from. The task cost H1 measures therefore includes pressure on the answer logits themselves, alongside the cost of carrying an anchored concept through the rest of the prompt. Any pull that covers the prompt has this property; we state it here so H1 is read as measuring both together.

In a full language model, the analogue of this label would pull the whole span between document boundaries, answer-like positions included, since nothing there marks a position as safe to exclude. So the prompt-only pull is a measurement instrument for this testbed, where the answer has a known confound; it is not a claim that the exclusion is needed. The unpulled answer position doubles as a first read on what a whole-span pull would change, and a whole-span variant is queued with the other labeling follow-ups.

Schedule and conditions

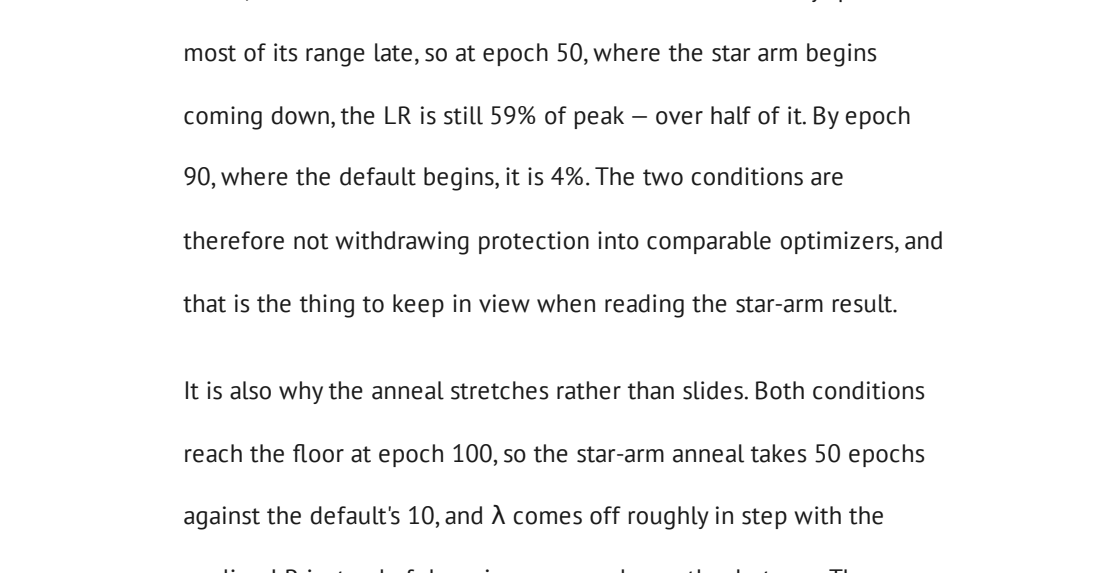
The LR schedule is the one from ex-2.1.3: 100 epochs, 10 warmup, cosine to 1%. The anchor weight ramps to its peak alongside the LR warmup, holds at peak through epoch 90 (by which point the LR has decayed to a few percent of peak), then anneals to a 10% floor at epoch 100. It never reaches zero, which is the lesson of ex-2.9.3 applied as issue #10 prescribes.

Both moves are minimum-jerk — the quintic that starts and ends at rest — rather than linear. That is what M1 used: the ex-2.9.x dopesheets take mini's default interpolator, which is `minjerk`, so every regularizer ramp and anneal in those experiments had this shape.

Whether the smoothness earns anything is untested; M1 never compared linear against smooth. It did test a *stepped* anneal, which worked only when the LR warmup restarted from zero at each step, so the one discontinuity anyone measured needed an accommodation to survive. A kink is the milder version of the same event (a jump in the derivative of λ rather than in λ itself) and may well cost nothing. We take the M1 default because this experiment has no budget to find out, and record the assumption here rather than in the code.

The sweep conditions are peak anchor weight $\lambda \in \{0,\ 0.03,\ 0.1,\ 0.3\}$ crossed with seeds $\{0, 1, 2\}$, so 12 runs. (λ is M1's symbol for a regularizer weight, Ω being the regularizer itself.) The $\lambda{=}0$ condition is the in-experiment control: same corpus, labels ignored. The other rungs chart the dose–response.

One **star arm** joins them: $\lambda{=}0.1^{*}$, identical except that the anneal starts at epoch 50 rather than 90, still reaching the floor at 100. Star arm is our shorthand, borrowed from the *star points* of a central composite design: it moves one factor off the scoring rung instead of crossing the whole grid, so it costs 3 runs rather than 12 and answers a single question. The question here is whether the long hold at peak buys alignment or just leaks: the anchor keeps pulling every labeled line for 90 epochs, and leakage is the secondary measurement we most expect to be sensitive to that. Against it stands the lesson of ex-2.9.3, that withdrawing protection while the optimizer is still hot lets the task loss reclaim the axis, and epoch 50 is a good deal hotter than epoch 90 (see the figure below). Spreading the withdrawal over the remaining 50 epochs, and stopping at a floor rather than zero, is what makes it worth trying. It is an unscored diagnostic; if it shows lower leakage at equal alignment, the anneal window becomes a design question for a later experiment.



Learning rate and anchor weight over training, each as a fraction of its own peak — the LR peak is fixed at 1e-2, the anchor weight's is the swept λ. The anchor weight's ramp and anneal are minimum-jerk; both anneals end at epoch 100 on a floor of 10% rather than zero, so the star arm's is the longer and gentler of the two. Dots mark the learning rate at the moment each anneal begins.

Drawing the two together settles something the prose was vague about, and not in the direction we assumed. A cosine decay spends most of its range late, so at epoch 50, where the star arm begins coming down, the LR is still 59% of peak — over half of it. By epoch 90, where the default begins, it is 4%. The two conditions are therefore not withdrawing protection into comparable optimizers, and that is the thing to keep in view when reading the star-arm result.

It is also why the anneal stretches rather than slides. Both conditions reach the floor at epoch 100, so the star-arm anneal takes 50 epochs against the default's 10, and λ comes off roughly in step with the cooling LR instead of dropping away underneath a hot one. The alternative — a fixed 10-epoch window sliding to 50–60 — reaches the floor while the LR is still around 42% of peak, which is close to the setting that produced M1/Ex-2.9.3's late collapses. That would be a fine stress test of the mechanism and a poor candidate for a schedule we might adopt, and the star arm is here to propose a schedule.

The moved factor is the *start* of the anneal, with the endpoint pinned to the end of training. The rate follows from those two, so the design has one knob.

Scoring

Hypotheses resolve on the middle rung, $\lambda{=}0.1$; the numbers in the results sections come from that rung. Call a rung **task-clean** if its `named_holdout` exact match is within 0.02 (absolute) of the seed mean of the $\lambda{=}0$ control. H1 asks whether *every* anchored rung is task-clean.

H2 claims that anchoring works, and that claim shouldn't fail just because we guessed a coefficient wrong. So its gates ask whether *some* task-clean rung clears the threshold: a rung can carry H2 even if H1 fails because a different rung broke the task. The task-clean requirement keeps this from being cherry-picking: a rung earns nothing by reaching alignment if it broke the task getting there. And the ladder has only four rungs, so the multiple-comparisons cost of reading across it is small. If the rung H2 resolves on is not $\lambda{=}0.1$, we say so and report both. If more than one other task-clean rung clears the gates, H2 resolves on the one with the highest m_{op1}; the choice is mechanical rather than ours. If no rung is task-clean, H2 fails on the gate it is scored against, and we report the alignment numbers anyway. The star arm is not a rung: it is reported alongside the ladder, but it takes no part in the "every anchored rung" test of H1 or in the search for a rung for H2 to resolve on.

Every hypothesis except H4 is scored on the final checkpoint of each run; H4 is the one that reads the trajectory, and its end-of-training value is that same final checkpoint.

H3 is scored on the same rung H2 resolves on, and only if H2(a) passed there. That restriction matters because the H3 gate is a ratio of margins. If both margins are at noise level, a 2× ratio between them would look like condensation where nothing condensed. H4 applies to every anchored run whose alignment ever clears a floor of 0.2. Where the anchor never took, the running maximum is itself at noise level, and a ratio of noise-level margins says nothing about stability. Runs below the floor are reported but not scored.

Measurements

Behavioral: exact match, NLL,[2] and the distance metrics of ex-2.1.3 on all eval sets, with the usual nulls from `sca.baselines`.[3] Exact match is the coarser scoring of the two: it only counts the argmax. So a run can hold accuracy while NLL drifts, and that drift is the earlier warning that capacity is being spent elsewhere.

Alignment maps: $\cos(h_{\ell,t}, \hat{v}_{\text{red}})$ on the alignment probe set, per layer × position. Write $\alpha_c(\ell, t)$ for the mean of that cosine over probe lines whose first operand is color c. H2, H3, and H4 all read one statistic off this map, the alignment margin

$$m(\ell,t) = \sum_c u_c\,\alpha_c(\ell,t) \;-\; \frac{1}{216}\sum_c \alpha_c(\ell,t), \qquad u_c \propto \texttt{redness}(c)^8,\ \textstyle\sum_c$$

In words: take the mean alignment weighted by label affinity, and subtract the plain mean over all colors. The margin measures how much closer a color sits to the anchor for being the kind of color the anchor pulls. The weights u_c are the label probabilities themselves, normalized, so the statistic asks about precisely the lines that felt the anchor term. A margin of 0 means the anchor axis is indifferent to redness. Since m is a difference of cosines, it runs from -2 to 2, but the ends are out of reach: 2 would need the red-weighted colors to sit exactly on the anchor while the cube average sits at the exact opposite direction. In a 64-dimensional stream, unrelated directions sit near $\cos = 0$, so the unweighted mean should stay close to zero, and complete success looks like $m \approx 1$.

▶ **Why weight by redness rather than group colors…**

H2 scores the layer mean at op1,

$$m_{\text{op1}} = \frac{1}{L+1}\sum_{\ell=0}^{L} m(\ell, \text{op1})$$

We average over anchored sites because the anchor applies to all of them equally, so there is no site selection to make. Alongside the mean we report the min and max over layers as unscored diagnostics; a wide min–max gap tells us which depths resist the anchor. H3 compares the same layer mean across positions, and H4 tracks m_{op1} over training.

Grading: for each op1 color, we take the layer mean of α_c at the op1 position and plot it against the redness of that color. This is the same quantity that m_{op1} summarizes, shown per color instead of contracted to a number. We read two statistics off this plot: Spearman ρ against `redness`, and Pearson R^2 against `sim¹·⁵`. Here `sim` is the angular similarity-to-red used by the intervention scoring in M1 (`sca.colorcube.sim_to_red`, restricted to this grid). The exponent comes from mapping the M1 ablation result from damage to alignment: damage graded as `sim³` and is quadratic in alignment, so alignment should grade as `sim¹·⁵`. H2(b) reads both statistics.

Leakage (secondary, no threshold): we fit a ridge probe for redness on the residual stream at op1, one probe per layer, after projecting out the $\hat{v}_{\text{red}}$ component. A high R^2 here would mean red is also encoded somewhere other than the anchor axis; we expect it to be low. R^2 is read on held-out colors, using a 5-fold split over the 216 op1 colors, so no color is scored by a probe that saw it. The ridge penalty is chosen within each training fold by an inner cross-validation, so no setting is picked by looking at the number we report.

Trajectories: every ~50 steps we record m_{op1} and validation loss, along with the LR and anchor-weight schedules. This is the apparatus from ex-2.9.3, carried over as the first queue item of issue #10. A flat loss curve does not mean learning is done, so the alignment trajectory is the evidence to consult before any decision to shorten the schedule.

Hypotheses

Unless stated otherwise, results are reported on the $\lambda{=}0.1$ condition (seed-averaged); the other rungs are context. Per the scoring rule above, the H2 gates read *some task-clean rung*, and H3 follows H2.

H1. Anchoring does not harm the task. Every anchored rung is task-clean: `named_holdout` exact match within 0.02 (absolute) of the seed mean of the $\lambda{=}0$ control. Partial: the gate passes at $\lambda \leq 0.1$ but fails at 0.3. The star arm is reported in the same table but scored only descriptively, per the scoring rule.

H2. The concept lands on the anchor, and does so in a graded way, at some task-clean rung. (a) $m_{\mathrm{op1}} \geq 0.5$. (b) Redder colors sit closer to the anchor, on either of two pre-named notions of *redder*. The grading statistic for a color c is the layer mean of α_c at op1, averaged over seeds before anything is ranked or correlated; over the 216 colors, it clears at least one of Spearman $\rho \geq 0.8$ against `redness`,[4] or Pearson $R^2 \geq 0.8$ against `sim`$^{1.5}$.[5] (c) The control condition shows $|m_{\mathrm{op1}}| \leq 0.1$.

Partial: (a) and (c) hold but (b) fails on both tracks, with the grading figure showing a step rather than a grade. In a step, the reds all sit at one alignment, the non-reds at another, and there is no ordering within either group.

Why (b) is two tracks, and how the gates were placed

This gate went through several review rounds before freezing, so the reasoning is recorded here in full, with the response shapes we checked and what each scores.

The shape of the pull itself fails both tracks, and that is the intended reading. The direct pull is proportional to `redness`^{8}, and a response that stayed proportional to it cannot clear a rank gate. Unrelated directions in a 64-dimensional stream put a floor of about 0.056 on the measured α_c (a spread of $1/\sqrt{64}$ per cosine, averaged over the 5 layers), or ≈0.032 after the three-seed mean the gate is scored on, and 129 colors have expected pull below 10^{-6}. So most of the cube ranks at random; simulation at that floor puts ρ near 0.21, and its R^2 against `sim`$^{1.5}$ is 0.44, far under the gate on both tracks. A response confined to the pull means the anchor caught the labeled tokens rather than *red*, which is the memorized-exemplar outcome (b) exists to catch. M1 sets the expectation for success: with labels drawn from this same affinity, its measured damage graded as low powers of `sim`.

The two tracks name two families a generalized response could take, one from each side of the milestone. Track 1 is ρ against `redness`. It accepts a response rising with the label-generating variable itself: a linear `redness` response scores ≈0.98 at the seed-mean noise floor. Track 2 is R^2 against `sim`$^{1.5}$, M1's ablation result mapped from damage to alignment (damage ∝ `sim`^{3} and damage quadratic in alignment give alignment ∝ `sim`$^{1.5}$; the suppression result maps to `sim`^{1} the same way). The 3/2 power also maximizes coverage of the family: $R^2 \geq 0.86$ against every power of `sim` from 1 to 3, and `redness`^{3} scores 0.91 noise-free, ≈0.86 at the floor. Each track is robust where the other is marginal. `sim` orders the cube by hue rather than by the `redness` formula, and it takes only 23 distinct values on this grid, so a sim-family response of any power scores ρ ≈ 0.76 on track 1 at the seed-mean floor, under the gate; its noise-free ceiling of 0.82 sits just over it, because midranks credit the ties. In the other direction, a linear `redness` response scores $R^2 = 0.81$ on track 2, right at the gate, while clearing track 1 with room to spare. Passing on either track accepts every graded shape above; requiring both tracks would fail most of them.

The floor is looser than the first draft of this gate assumed, and the numbers above are the corrected ones. The earlier estimate divided the per-cosine spread by 27 probe lines as well as 5 layers, which the measurement cannot do: at op1 the 27 lines of a color share a state, as the method notes. Correcting it (0.011 → 0.056 per seed) leaves both thresholds where they were frozen and changes what each track covers. Track 2 now carries the graded shapes that are not linear in `redness`: `redness`^{3} scores ρ ≈ 0.70 on track 1 against $R^2 \approx 0.86$ on track 2, and the `sim` family likewise. Track 1 carries the `redness`-linear family. A response at half amplitude — roughly the least H2(a) accepts — clears neither track unless it is close to linear in `redness`, so the gate is stricter than we first estimated rather than looser.

A pure step clears neither gate. The measured alignment is continuous, so a step orders the colors within each of its two levels arbitrarily. Its expected ρ is at most 0.75 over all placements (a step at redness 0.5, the memorized-exemplar shape, scores 0.34), and its R^2 against `sim`$^{1.5}$ is at most 0.73 over placements in either variable. The margins between those ceilings and the 0.8 gates are a few hundredths, which is one more reason the figure stays in charge. What would carry a steppy response over a gate is residual ordering within the levels, which is some grading after all; for shapes in between, the arbiter is the windowed mean drawn through the scatter.

H3. The concept condenses at the position that carries it: that is, m_{op1} is at least 2× the same layer mean at each of `+`, op2, and `=`. If the margin at a comparison position is zero or negative, the 2× test has no content, so we treat it as met. (H3 is scored only when H2(a) has already put m_{op1} above 0.5, so a comparison site at or below zero is dominated outright.) The anchor pulls those four positions identically, so the comparison is between sites that differ only in what the task does with them. The answer and newline positions are outside the pull; they are reported but not scored. Per the scoring rule, H3 resolves on the rung H2 does, and only if H2(a) passed there.

We also anticipate a specific alternative outcome: broadcast, where all positions of a labeled line move toward $\hat{v}_{\mathrm{red}}$ roughly equally. That would fail H3 while H2 passes. It would tell us something about sequence-level labeling rather than about anchoring itself; a logsumexp position-pooling variant is the queued response.

H4. The late-instability mechanism of ex-2.9.3 does not reappear under the anneal-to-floor schedule. Concretely: for every anchored run whose running maximum of m_{op1} reaches 0.2, the end-of-training value is at least 0.8× that maximum. The 0.2 floor keeps the gate from resolving on noise. In a simulation of unrelated 64-d states, the per-checkpoint noise in the margin is near 0.021, the maximum of a noise-only trajectory is near 0.06, and a ratio of noise-level margins fails the 0.8× gate about as often as not. The floor is therefore about 3× a noise-only trajectory's peak, and well under H2(a)'s 0.5. Runs below the floor are reported but not scored. Partial: violations confined to the $\lambda{=}0.3$ condition.

Task cost (H1)

condition	seen EM	holdout EM	holdout NLL	`open` dist	Δ holdout EM	H1 gate
ex-2.1.3 `v216`	1.000	0.995	0.025	0.141	—	reference
λ = 0	1.000 ±0.000	0.999 ±0.002	0.019 ±0.009	0.133 ±0.002	+0.000	clean
λ = 0.03	1.000 ±0.000	0.997 ±0.004	0.023 ±0.012	0.137 ±0.010	-0.001	clean
λ = 0.1	1.000 ±0.000	0.997 ±0.004	0.017 ±0.004	0.130 ±0.003	-0.001	clean
λ = 0.3	1.000 ±0.000	0.996 ±0.004	0.024 ±0.013	0.136 ±0.004	-0.003	clean
λ = 0.1*	1.000 ±0.000	0.997 ±0.002	0.021 ±0.005	0.135 ±0.003	-0.001	clean

Behavior by condition. Each cell is the seed mean, with half the seed range beside it. EM is exact match on the answer token. NLL is the surprise of that token in nats. `open` dist is the RGB distance from the guessed color to the true mix, on pairs with no named answer. Δ holdout EM is the signed gap to the λ = 0 control, and the last column reports the H1 gate, which passes when that gap is within 0.02. The first row is the published ex-2.1.3 `v216` condition; it is an external reference, not part of this sweep.

H1 holds, with room to spare. Every anchored rung is task-clean. The largest gap to the control on `named_holdout` exact match is 0.0026, against a gate of 0.02, and that includes $\lambda{=}0.3$. Seen pairs are at 1.000 everywhere.

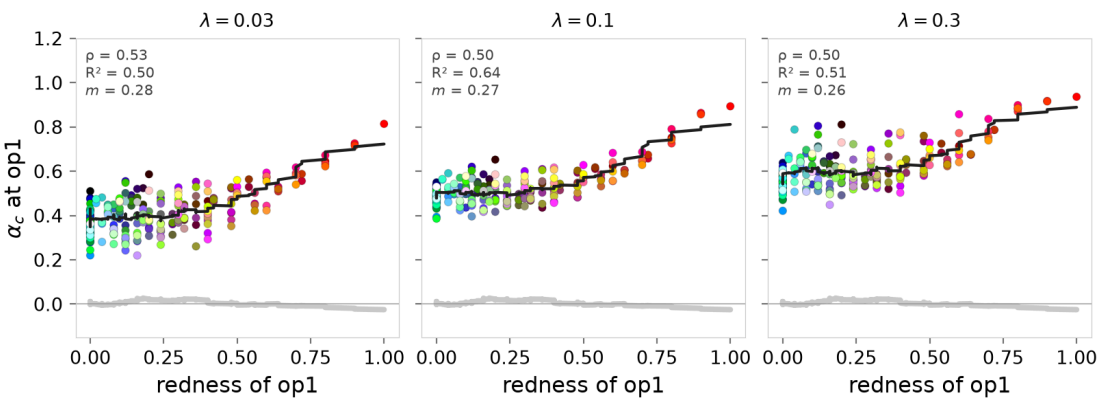
The finer scoring agrees. Answer NLL stays between 0.017 and 0.024 nats with no trend in λ. So there is no sign that the anchor is quietly consuming capacity while the argmax stays correct.

The `open` -pair distances sit together near 0.134. That is above the floor of 0.086, and also above the 0.115 expected of a guesser that flips a coin between the two names bracketing the true mix — so on this metric the models do not quite reach that null, though they are far under the 0.426 of a guesser that ignores the prompt. The shortfall is in the last step of resolving between two adjacent names, and it is a property of this testbed rather than of the anchor: the control sits with the rest, and this is the same picture ex-2.1.3 reported for this condition.

So at these weights, the dose–response curve has no cost side. The next section looks at what the anchor buys.

Alignment and grading (H2)

The concept moved onto the anchor, but so did everything else. The figure below shows the per-color response at op1. The whole cube lifts off the control baseline together, and the reddest colors sit only a little above the rest.



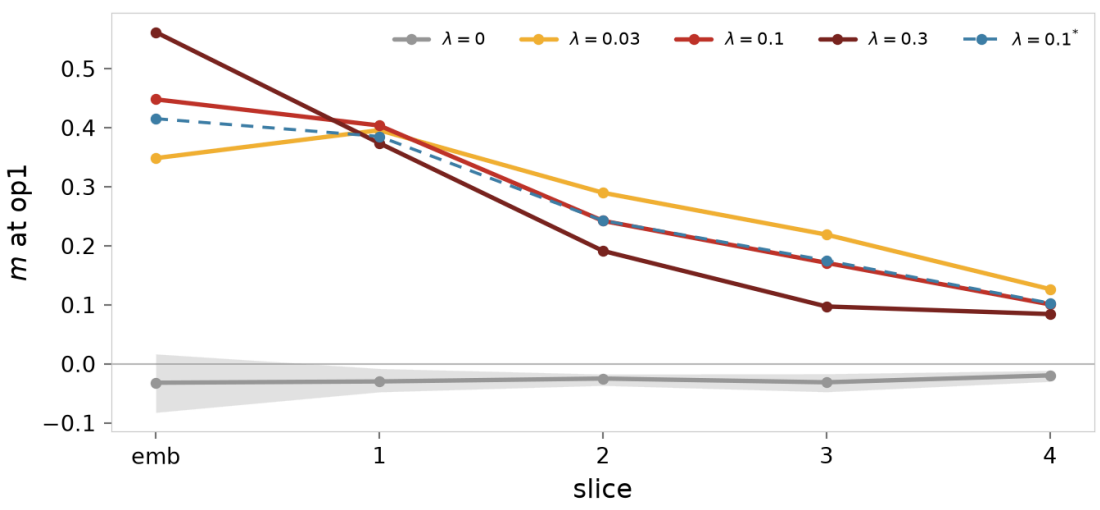
Alignment at op1, per color: α_c, the layer mean of $\cos(h, \hat{v}_{\text{red}})$ averaged over seeds, against the redness of the color. One mark per color, drawn in that color; the heavy line is a 25-color sliding mean over the redness ordering, and the flat grey band underneath is the sliding mean of the $\lambda{=}0$ control. The grading statistics and m_{op1} are quoted per panel; both H2(b) gates sit at 0.8.

H2 fails on (a) and (b); (c) holds. No rung reaches the 0.5 margin: m_{op1} is 0.28, 0.27, and 0.26 at $\lambda = 0.03, 0.1$, and 0.3, and the seeds agree to within 0.019. The hypothesis only needed *some* task-clean rung to clear the gate, and since every rung is task-clean, the search had the whole ladder to work with; none of it clears the gate. The grading statistics land in the same place: the best of them is $R^2 = 0.64$ at $\lambda{=}0.1$ against a gate of 0.8, with $\rho \approx 0.51$ on the rank track. The control clears (c) comfortably: $|m_{\text{op1}}| = 0.027$, well inside 0.1 and about the size of the per-checkpoint noise (0.021). So without the anchor, redness does not sit preferentially on this axis. Note the margin asks a narrower question than decodability: a probe can still read redness off the untouched stream, as the leakage measurement below shows. The margin only asks whether this one direction singles red out.

Why does the margin stay low? The figure shows it directly: the whole cube moved. Mean alignment at op1 goes from 0.008 in the control to 0.42, 0.53, and 0.62 up the ladder. That is a color-independent shift that climbs with λ while the margin does not. The margin is a contrast between red and non-red colors, so a shift shared by all colors does not count toward it. The anchor term, though, is satisfied by exactly that shift. This is how the term can be well optimized while the hypothesis is unsupported.

Why is a shared shift available to the model at all? Because of what the pull can see. Whether a line is labeled is a coin flip the model cannot observe: nothing in the input distinguishes a labeled line from an unlabeled one. So the model can only respond to the *expected* pull. At op1 that expectation is proportional to the label affinity of the color at op1, which is red-specific. But at `+`, op2, and `=`, it is still proportional to the label affinity of *op1*, which says nothing about the token actually sitting at those positions; op2, for instance, is uniform over the cube. So three of the four pulled positions ask for a move that does not depend on their own token, and the model supplies one.

Alternatively, the whole cube moved because nothing prevents it. We saw this in M1, too: without other regularization terms to repel colors from each other or from a subspace, the model can satisfy the anchor signal by rotating everything in that direction.



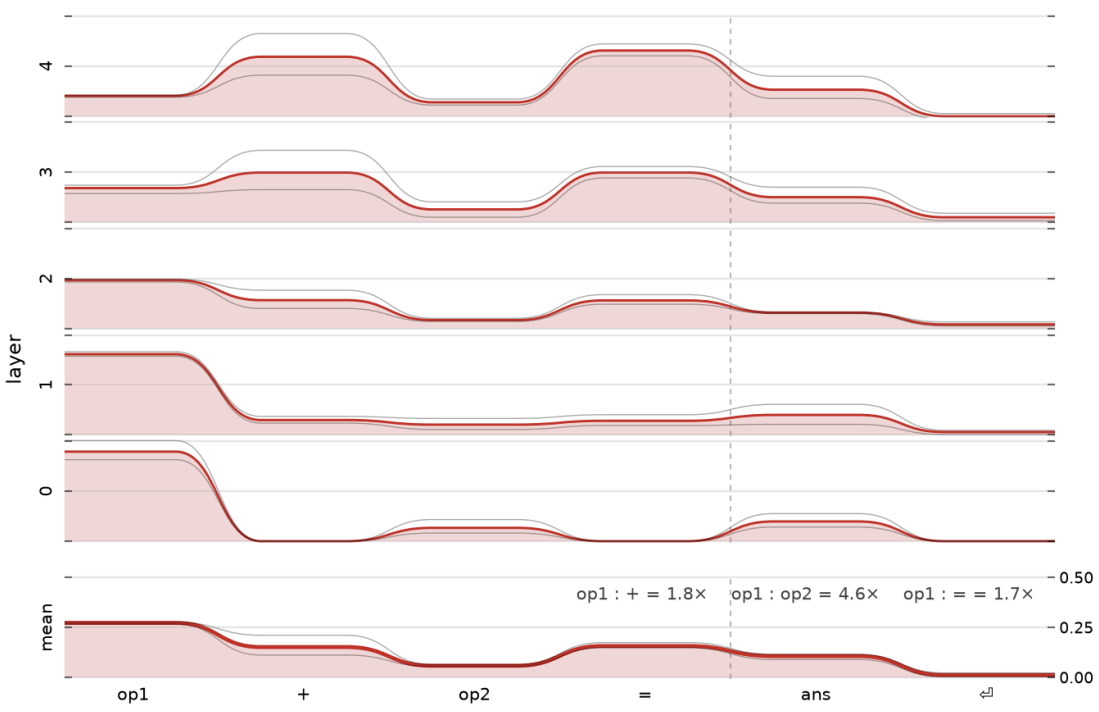
Where the margin lives in the stack: the alignment margin at op1 by layer — the per-depth terms whose mean is m_{op1}. Depth 0 is the token embedding, depth 4 the last block's output. Lines are seed means; the shaded band around the control is its seed min–max, as the scale of a null. The anchor pulls all five layers equally, so the slope comes from the network, not from the pull schedule.

The per-depth view shows where what little selectivity exists is located, and it is not a sag in the last layer. At every weight, the margin peaks in the embedding or the first block and falls away with depth. Raising λ steepens that decay rather than lifting the curve: at $\lambda{=}0.3$ it runs 0.56 → 0.08 from embedding to last layer, against 0.35 → 0.13 at $\lambda{=}0.03$. So a heavier pull buys selectivity in the token embedding, then loses it again through the stack. That fits the idea that the deeper layers are where the shared, color-independent shift is cheapest to produce: the embedding is shared across positions, so moving it moves op2 too, while a block's output can depend on position and context.

A second reading is that the deeper layers are committed to next-token prediction. By the last block, the state at op1 has to emit `+`, and this model is probably deeper than the task needs. The character-level testbeds of ex-2.1.5 did put their concept readout deeper in the stack, but there it peaked at the pre-answer position, after the color had been named. That precedent does not transfer to op1, where the color has only just arrived.

Condensation vs broadcast (H3)

H3 is not scored. The scoring rule says it resolves on the same rung as H2, and only if H2(a) passed there. H2(a) did not pass at any rung. We still report the map below: the condensation question was about the position profile, and even a low margin has a shape worth looking at.



Margin by position and depth: $m(\ell, t)$ across the six positions of a line, on $\lambda{=}0.1$, seed mean. One panel per layer, and the seed mean over layers in the bottom panel, drawn heavier. The shaded area runs from zero to the seed mean; the hairlines are the seed minimum and maximum. Positions are ordinal, so the risers are sloped, but nothing is measured between them. The four pulled positions sit left of the dashed rule; the answer and newline are measured only.

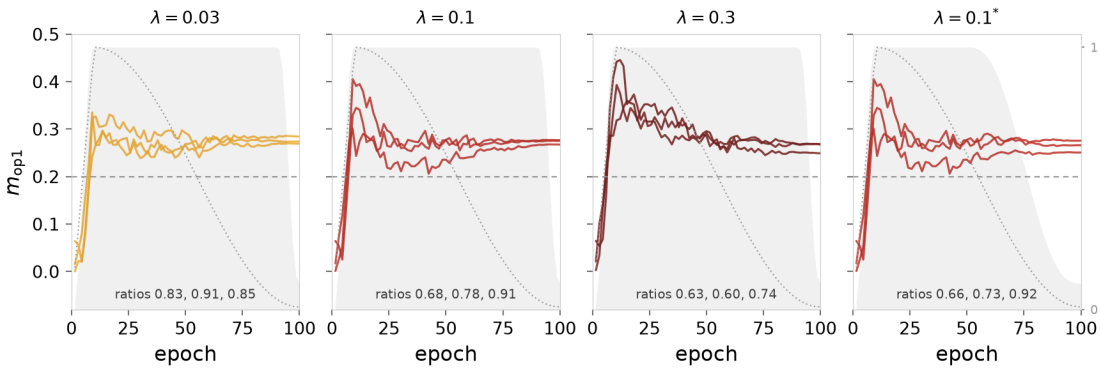
op1 does lead, but not by the factor H3 named: 1.8× against `+`, 1.7× against `=`, and 4.6× against op2. Had the gate been live, the op2 comparison would have passed and the other two would not. So the outcome sits between condensation and broadcast rather than at either. The ordering itself is informative. op2 is the position where the expected pull carries no information about the token actually there, and it has the least margin. `+` and `=` are constant tokens, so their states are free to encode the context of the line; they carry 56% and 58% of the margin at op1.

The layer mean also hides an inversion that the panels above it show plainly. In the embedding and the first block, op1 is the only position with much margin at all (0.45, against 0.00 at `+`). By the last block the ordering has reversed: op1 sits at 0.10, while `+` and `=` carry 0.30 and 0.33. So *red* enters at the token that denotes it, and with depth it moves to the positions that follow it in the line. That is what a model with attention would do with a fact about the line, and it is not a distinction the condensation/broadcast pair anticipated. Whether that pattern is worth reserving an axis for is an intervention question, which this experiment does not answer.

The answer position, which the pull never touches, still shows 0.11, about 39% of the margin at op1. That is the confounded quantity the method set aside: a red op1 drags the mixed color toward red, so part of this margin is spillover from an anchored operand, and part is the redness of the answer itself finding the axis. The newline sits at 0.01. A position that neither carries a color nor feels the pull should sit near zero, so this is a useful check that the map is not simply warm everywhere.

The method also asked us to check the last layer at `=`. The margin there does not dip relative to the earlier layers: it is 0.33, against 0.07 at layer 1. So we see no sign of the answer logits pushing back on the pull at that position. With the task cost of H1 at zero, that is the consistent reading.

Training stability (H4)



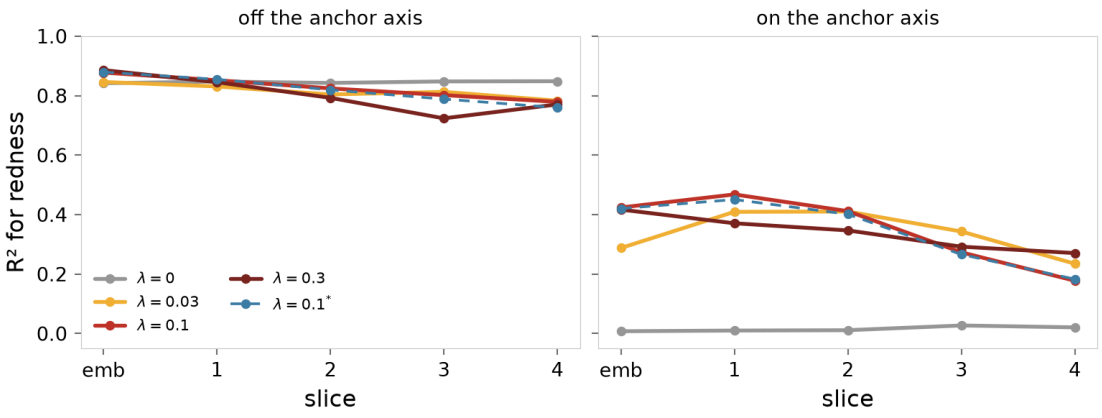
Alignment over training: m_{op1} measured every 50 steps. One panel per condition, one line per seed. The pale band behind is the anchor weight as a fraction of its peak, so its descent marks the anneal window; the dotted line is the learning rate. Both use the right-hand scale, numbered on the last panel. The dashed horizontal rule is the 0.2 floor of H4; a run below it is reported but not scored. Ratios are the end-of-training value of each seed over its running maximum.

H4 fails, but not in the way it was written to catch. All 12 anchored runs clear the 0.2 floor, so all of them are scored. 7 of them end below 0.8× their running maximum, spread across 3 of the four anchored conditions, including $\lambda{=}0.1$. The partial outcome (violations confined to $\lambda{=}0.3$) does not apply. The control runs peak at 0.064. That is under the floor and in line with the 0.06 a noise-only trajectory reaches, so they are reported but not scored, as designed.

The trajectories show a slide rather than a collapse. Every anchored run rises through the LR warmup to a peak between epochs 9 and 17, then loses about a quarter of that over the following forty or so epochs and holds flat from there. The anneal is not the trigger. The loss of margin is complete long before epoch 90; the earlier anneal in the $\lambda{=}0.1^*$ arm does not move it; and the end value is the same (0.26–0.28) across a tenfold range of λ. So we are not seeing the ex-2.9.3 mechanism, where protection is withdrawn while the optimizer is still hot and the task loss reclaims the axis. The schedule did the job it was designed for; the margin settles at a level the pull does not set.

Read together with the growing mean alignment of H2, the slide looks like the drift catching up. The peak reflects early selectivity, and what probably erodes it is the rest of the cube arriving on the axis afterwards.

Secondary measurements



Where redness is readable at op1. Left: held-out R^2 for a ridge probe that predicts redness from the residual stream at op1, with the $\hat{v}_{\mathrm{red}}$ component removed first. This measures how well red can still be read from the other 63 directions. Right: the same target predicted from the anchor component alone, as squared correlation. Both panels show per-layer seed means. The sample is the 216 op1 colors, with a 5-fold split so no color is scored by a probe that saw it during fitting. Neither measurement has a pass/fail threshold.

Leakage is essentially undiminished. Project out the anchor direction, and redness is still readable from the remaining 63 directions: $R^2 \approx 0.83$ on $\lambda{=}0.1$, against 0.85 for the control. The difference is a few hundredths, concentrated in the deeper layers. Meanwhile the anchor axis itself goes from carrying 0.02 of the redness variance to 0.35. So the axis became readable for redness, but redness did not leave anywhere else. Calling that a second copy of the concept would be too strong; the exploratory section below shows why.

Does this leakage bound what an intervention on the axis can do? That question is narrower than it looks, and the exploratory section takes it up. Redness is a function of color, and a model that answers `red + blue = purple` has to represent color; so a high off-axis score may reflect the color cube rather than a second copy of the concept. What the number does establish is that anchoring did not *concentrate* redness onto the axis at the expense of elsewhere. An anti-subspace term would constrain exactly this quantity, by pushing the unlabeled cube *off* the axis. Without it, nothing holds either the leakage or the undifferentiated drift of H2 in check.

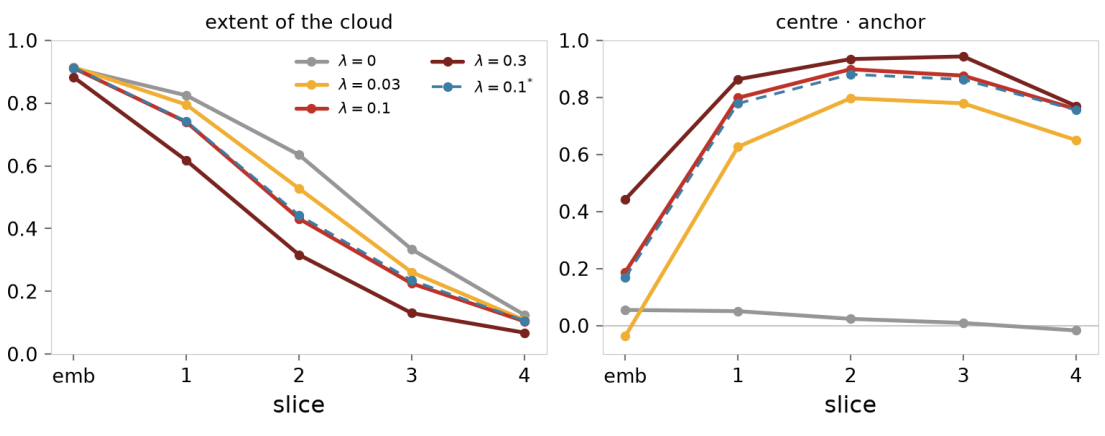
The star arm changes nothing measurable. The $\lambda{=}0.1^*$ arm tracks $\lambda{=}0.1$ on every measurement: m_{op1} of 0.264 against 0.273, off-axis R^2 of 0.82 against 0.83, and holdout accuracy identical to three decimals. The hypothesis behind this arm was that a long hold at peak anchor strength is what leaks. Cutting forty epochs from the hold leaves the same leakage, so on this evidence the hold length is not the knob. Also, the two arms are the same run up to epoch 50, and their trajectories are still together at epoch 100; that doubles as a check that the pipeline is deterministic where it should be.

Exploratory analyses

Everything below is post hoc: we conceived it after seeing the results, in response to two questions about the primary reading. It reads the published checkpoints rather than the preregistered measurements, and it scores no hypothesis.

Did the cube collapse, or did it swing? (post hoc)

The first question: M1 used two repulsive terms that this experiment left out, `separate` and `anti-subspace`. Which one are we missing? In M1, without `separate` to keep near-duplicates apart, the latent cube ended up in a small region of the sphere. The two terms predict different shapes here. If the anchor squeezed the 216 colors together, `separate` is the missing term; if it moved them as a body onto the axis while they kept their extent, `anti-subspace` is.



Where the color cloud is, and how big: the shape of the 216 op1 states, seed means, per layer. Left: the extent of the cloud, the mean squared distance of a color from the centre (states are unit-norm, so this runs from 0 for a collapsed cloud to 1 for a spread one). Right: the cosine between that centre and the anchor direction — where the cloud sits, as opposed to how big it is. In the control that value is the scale of an unrelated direction in 64 dimensions.

The answer is mostly *swing*, and the surprise is in the control. Even un-anchored, this model concentrates the color cloud sharply with depth: its extent falls from 0.91 at the embedding to 0.13 at the last layer, so by the top of the stack the 216 colors already sit within a small cap. The task itself causes this: the state at op1 is being turned into a prediction, and the prediction is the same token (`+`) for every color. So whatever `separate` would protect against here, the network arrives at without help.

Against that baseline, the anchor changes position far more than size. In the control, the centre of the cloud sits 0.02 of the way onto the anchor, which is the scale of an unrelated direction. With the anchor on, it reaches 0.90 at $\lambda{=}0.1$ and 0.93 at $\lambda{=}0.3$. Meanwhile the extent at the same layer only goes 0.64 → 0.43 → 0.32. So at the scoring rung the cube kept about 68% of its extent and swung as a body onto the axis. That is the shape a missing anti-subspace term predicts: nothing was constraining the *common* component of the colors, and the common component is what moved.

There is real compression at the top of the ladder, though. At $\lambda{=}0.3$ the mid-stack extent is down to 50% of control, and the trend across rungs is monotone. So a `separate`-style term would start to matter at a weight beyond what this experiment tried. On the present evidence, `anti-subspace` is the term to reach for first, with `separate` second; and by pushing unlabeled colors off the axis, the anti-subspace term would relieve some of the same pressure.

Is the off-axis leakage about *red*? (post hoc)

The second question: is "an intervention would leave a copy behind" the only reading of the leakage number? It is not, and the check is cheap: run the same off-axis probe for colors the anchor never touched. Redness is a function of color, and a model that answers `red + blue = purple` must represent color. So redness might be recoverable off the axis for a reason that has nothing to do with anchoring.

	off the anchor axis		on the anchor axis	
target	**λ = 0**	**λ = 0.1**	**λ = 0**	**λ = 0.1**
redness	0.846 ±0.059	0.827 ±0.078	0.015 ±0.031	0.351 ±0.184
greenness	0.818 ±0.032	0.836 ±0.033	0.016 ±0.037	0.067 ±0.082
blueness	0.823 ±0.051	0.836 ±0.037	0.005 ±0.008	0.057 ±0.062
R	0.975 ±0.008	0.938 ±0.052	0.076 ±0.072	0.146 ±0.143
G	0.951 ±0.046	0.939 ±0.031	0.058 ±0.088	0.083 ±0.121
B	0.961 ±0.017	0.947 ±0.034	0.034 ±0.048	0.074 ±0.063
sim$^{1.5}$	0.685 ±0.116	0.751 ±0.130	0.010 ±0.026	0.481 ±0.194

What is recoverable from the op1 residual stream, per target. Each cell is the seed mean, with half the seed range beside it. Rows are the property being predicted: redness; the same formula rotated onto the green and blue corners; the raw channels; and the sim$^{1.5}$ shape that H2(b) scores against. The left pair of columns is held-out ridge R^2 with the anchor direction removed, averaged over layers. The right pair is squared correlation with the anchor component alone. Reading down a column compares redness against targets the anchor never pulled.

Off the axis, redness is not special. On the anchored model, greenness scores 0.84 and blueness 0.84, against 0.83 for redness, and the raw channels score around 0.94. The control looks the same. So the off-axis number reflects the color cube: every corner of it is recoverable, because the task needs it to be, and redness is just one function of the colors among many. It is not evidence of a second, red-specific encoding.

On the axis, redness *is* special; this part of the primary reading survives. The anchor component carries 0.35 of redness variance, against 0.07 for greenness and 0.06 for blueness, up from 0.02 in the control. The anchor put red where we asked for it (and rather more of `sim¹·⁵`, at 0.48), and nothing else came with it.

What this costs the primary reading is the sentence about intervention. The leakage measurement says a linear probe can still recover redness after the axis is removed. It does not say the *model* would still behave red-ly; those are different claims. The induced error in M1 was measured on the model's own output under an edit, not on a probe. So the surviving claim is narrower: anchoring here added a red-selective direction without concentrating red onto it. What an edit to that direction would do to behavior is a question this experiment cannot answer; it is the reason to run the intervention, not a prediction of its result.

▶ **The candidates the skeleton anticipated…**

Discussion

Summary of results. H1 passed; H2 and H4 failed. H3 was not scored, because its frozen precondition, H2(a), did not hold at any rung. The anchor is free: no rung cost measurable accuracy, NLL, or open-pair distance. And it does move the residual stream a long way onto the chosen direction. What it did not do is move *red* there selectively. The margin settles near 0.27 against a gate of 0.5, and the best of the grading statistics reaches 0.64 against 0.8. Both are flat across a tenfold range of λ, while the color-independent part of the response climbs with it.

A mechanism that fits. The term is selective about which *lines* it fires on, but says nothing about *positions*. It fires only when op1 is red enough to draw a label, so the pull a color receives really is red-specific; that part of the design worked. But on a labeled line, the term asks all four prompt positions to move the same way, and three of those positions hold tokens whose own color the label says nothing about. op2 is the clear case: on labeled lines its color is uniform over the cube, just as on any other line, so no function of the token sitting there can satisfy the pull. What can satisfy it, at every position at once, is a shift that ignores the token entirely. That shift costs the task nothing measurable here, and it is what we observe. Several results fit this reading: the margin stays flat in λ while the mean does not; the margin is largest in the shared embedding and decays through the position-aware blocks; and the trajectory peaks early and then slides. One interpretation of the slide is that the selective response arrives first and the rest of the cube then follows it onto the axis. The trajectories are consistent with that, but a single design cannot separate it from the alternatives.

Implications for the M1 → M2 transfer. The M1 loss carried an anti-subspace term alongside the anchor, at a weight two orders of magnitude below it. This experiment deliberately left that term out, to ask whether one attractive term suffices. On this evidence it does not, and we come away with a candidate reason rather than a bare null: the anti-subspace term constrains exactly the quantity that ran away here. The exploratory section supports that reading over the alternative. The cube swung bodily onto the axis at the scoring rung while keeping most of its extent, which is a common-component problem, not the near-duplicate collapse that `separate` guards against. That section also shows the cloud concentrating strongly with depth on its own, anchor or no anchor. It would be too quick to read that as the task doing the work of `separate`, though: the last layer only has to tell the colors apart well enough to predict `+`, which it could manage from a compact cloud. Adding the anti-subspace term is queued as the direct test.

What we should *not* conclude is that an intervention would find a spare copy of red waiting. The leakage number says a linear probe can still recover redness with the axis removed, and the exploratory check shows that greenness and blueness score the same. So what survives removal is the color cube the task needs, not a red-specific second encoding. What an edit to the anchored direction would do to the model's behavior is a separate measurement. And projecting the whole axis out is only the crudest edit available; a suppression shaped to act on the red-selective part of the response might succeed where that one fails. Either way, the effect of an intervention is something to measure, not something to predict from here.

Open questions. We cannot yet separate "the bare term is insufficient" from "the *span* pull is what dilutes it", because this run changed both at once relative to M1. M1 labeled atomic samples, so every pulled position there carried the labeled thing. The queued position-pooling variant (logsumexp instead of a mean over positions) is the direct test. Pulling op1 alone would be cleaner still: it would make every pulled position label-informative while leaving the term itself unchanged. A smaller question the results raise is whether the drift is bounded by the task, or would keep climbing at larger λ.

Under H3 we saw a depth inversion: the concept enters at its own token and migrates to the positions downstream of it. This is most likely a transformer behaving as transformers do. For an intervention on *red* across the board it may not matter, since we would want the edit to apply at every (layer, position) site anyway. It becomes a live question at D2.3, which asks for suppression that degrades completion while leaving verification intact. That calls for the concept to be localized somewhere in the stream, and this run offers no evidence either way on whether an anchor can be confined to a chosen region of it.

A note on the schedule work: the star arm found nothing, and the H4 failure was a slide rather than the collapse seen in ex-2.9.3. So the anneal design carried over from M1 looks sound on this testbed. Two things the trajectories do raise are queued in `todo-science.md`. The first is the alignment decay itself, which is the H4 result read as a phenomenon rather than as a gate. The second is the length of the schedule: the margin is flat from roughly epoch 50, so a shorter run may cost nothing. That is a budget question, not a way to keep the early peak; the peak never cleared H2(a) either, and choosing a stopping point from this data would be selecting on noise.

---

1. A Bernoulli draw: one biased coin flip per line, independent of every other line. ↩
2. Negative log-likelihood: the surprise, in nats, of the probability the model assigns to the correct answer token, with 0 meaning certain and correct. ↩
3. Reference scores for degenerate predictors (the corpus-mean color, a uniform draw, the identity of op1), which make a distance number readable as good or bad. ↩
4. Spearman ρ is a correlation coefficient computed on ranks rather than values, so it measures whether the ordering agrees and doesn't care about the shape of the curve. ↩
5. The squared Pearson correlation: the share of variance in the response explained by a linear function of $\mathtt{sim}^{1.5}$. It is scale- and offset-invariant, so it tests proportionality to the shape M1 implies for alignment with no tuned constant. ↩