Ex 2.2.2: a designed response to suppressing *red*

> We add fallback control: a training term that teaches the blocks what to answer once the concept is gone, aiming at a designed fallback answer, with the concept's placement kept out of its gradient. Does the fallback appear, does the intervention stay effective and selective, and does seed variability fall? The fallback appears in every seed, costs nothing on the task and a little margin, and reaches the plain projection at more than half strength. The reflection it was trained at is destructive on non-red lines in every anchored model, with or without the term, because the syntax token embeddings carry the axis.

Findings

- The designed response (H1) — **holds.** Fallback accuracy on the red lines with a clean visible operand under `redirect`: 0.999, against 0.015 without the term; gate 0.8, and the margin above the reference clears the 0.025 floor.
- Task and placement intact (H2) — **holds.** Clean accuracy gap from the no-fallback condition: 0.0002 (gate 0.02). Margin at the end of training: 0.925 of the no-fallback value (gate 0.8).
- Selectivity (H3) — **partial.** Non-red deficit under `redirect`: 0.749 (gate 0.02; the no-fallback condition loses 0.774 under the same edit). Under `projection`: 0.114 against 0.024. That difference is smaller than the 0.124 resolution floor, so the second clause holds, but as an unresolved difference.
- Transfer from the antipode to zero (H4) — **holds.** Fallback accuracy at $\gamma = 1$ is 0.63 of its value at $\gamma = 2$ (gate 0.5), and the sweep has no dip.

> **How to read this draft**
>
> The hypotheses, their gates, and the method were frozen at commit `d72340e7`, before any run. Each hypothesis section opens with that frozen prediction, and its results follow in place.
>
> Two corrections landed after the freeze, each with a `REVIEW` note beside the text it changed. The first is the count of red lines with a clean visible operand, now 296 rather than 298, because one color sat on the 0.5 contour to roundoff. The second is the gradient paragraph in the method, which now accounts for the tied embedding.
>
> Anything conceived after seeing the data is under Exploratory analyses, marked as post hoc (E7 and E8).

Why this experiment

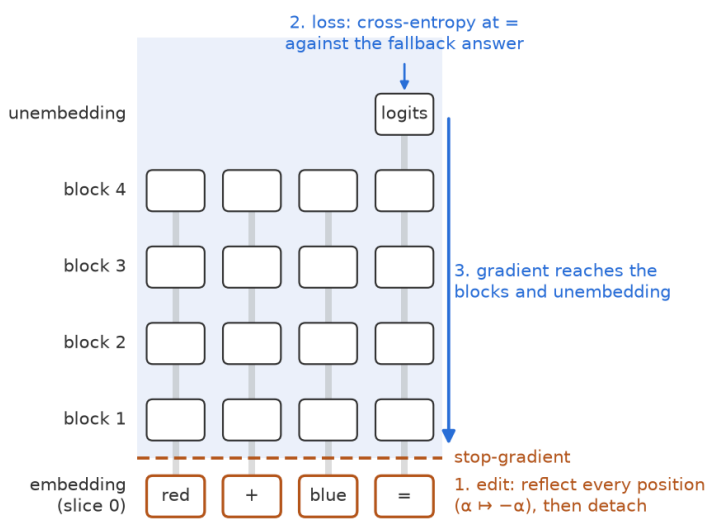
Ex-2.2.1 showed that projecting the anchor axis out of the D2.1 checkpoints suppresses *red*. Red-line accuracy falls from 1.0 to 0.09, in proportion to how red the line is, and stays inside the bound the placed geometry set.

A suppressed red line still decodes to a color, about half the time a one-step neighbor of the true mix. Which neighbor varies by seed: at least five of the nine seeds agree on 13% of red lines. Nothing in training said what a removed concept should decode to, so the answer is whatever the untrained region of the stream produces. M1 called that spoofing, and it is where the seed spread in ex-2.9.1 came from.

The remedy in M1 was fallback control (ex-2.9.2): one loss term, applied to the decoder only, that pins the antipode of the anchor axis to a designed null answer, mid-gray. Reflecting a state through the axis then collapsed the response to a tight cluster on the bound the null predicted, across 32 seeds. Under plain zeroing the term added little, and that report said a trained fallback should be paired with the redirect it was trained at.

Here we bring the term into the transformer, keeping the D2.1 grammar and recipe. This comes before the operation work in the D2.2 plan changes the grammar.

The term is one loss, but it touches the model in three places: the edit at the embedding, where every position is reflected through the axis and then held fixed; the loss, read at the `=` position of each red line; the gradient, which reaches the four blocks and the readout, and stops at the reflected states, so it cannot move the placement.



Where the fallback term acts. *One red line,* `red + blue =`*, as positions (columns) against slices (rows). The numbered annotations are the three places above; the blocks run forward from the edited embedding as usual, mixing positions through attention, and the dashed line is where the gradient stops.*

In M1 the decoder took a constant input, the antipode itself, so the term was decoder-only by construction. Here a stop-gradient[1] gives the reflected state that same role.

The fallback answer for a continuous concept also has to be chosen. We use the center of the operand-averaged null, which works out to be the visible operand mixed with *mid-gray*; the method derives it.

This addresses the third risk in the plan, that the response to suppression is undesigned. One mismatch carries over from M1: the response is trained at the antipode, while the ex-2.2.1 projection lands the state at zero. H4 measures how far the response transfers between the two.

The nearest published analogue is LUNAR (arXiv:2502.07218): one matrix edit after training, redirecting the activations of the data to forget into the model's own refusal region. LUNAR designs the response by choosing a region the model already produces; ours is trained at a state the model never otherwise visits, which is where the mismatch above comes from. E6 fits a LUNAR-style edit to the no-fallback checkpoints, so the two designs can be compared.

Natural language. The three parts all transfer to the natural language domain: the edit would be the reflection at the first anchored slice, applied at every position; the loss would be computed on the continuation (the tokens of the response), rather than at one `=` position; the gradient stops before the edit. The edit happens based on alignment of the states to the anchor, so it's grammar-agnostic and token labels are not needed.

The fallback answer still has to be chosen; some candidates: **(a)** a template continuation, so the training data is pairs of context and designed response; **(b)** the model's own refusal region, which is the LUNAR choice (and which could be anchored separately); **(c)** a true null, the continuation a reference model gives when it has never seen the concept, matched with a KL term.[2]

Spoofing in that setting is confabulation: a plausible answer drawn from the neighbors of the concept. The *mid-gray* answer here is a trained abstention that keeps the model on task (rather than, say, emitting a syntax token like `+` or `=` where the answer should be). A natural language version would want that same property.

▶ **Glossary…**

Conditions

Two of the three conditions are ex-2.1.10 checkpoints already in the store, re-scored here through the same code path as the new runs. The third is trained here.

condition	training	seeds	source
un-anchored	the recipe with the anchor weight at zero	3	ex-2.1.10 `lam0`
no-fallback	the D2.1 recipe: anchor + anti-subspace term	9	ex-2.1.10 `either-t100`
fallback	the recipe + anti-anchor term (0.1) + fallback term (w_fb = 0.05)	9	new

Every hypothesis scores against the fallback condition. The no-fallback condition is the reference for it under every intervention.

The arms below have three seeds each. We report the same statistics for them, without gates.

arm	what changes	question
`fb-only`	no anti-anchor term	does the fallback need the antipode hemisphere cleared for it?
`anti-only`	no fallback term	does the anti-anchor term alone move the response?
`fb-w0.01`, `fb-w0.25`	the fallback weight, a factor of five either side	is 0.05 on a plateau?
`recipe`	both new weights at zero	does the new code path reproduce the ex-2.1.10 checkpoints?

The interventions

All of these run through the projection operator of the eval contract: it removes a fraction γ of the e_1 component, then puts the state back on the sphere. At γ = 2 the operator is a reflection, flipping the component instead of shrinking it.

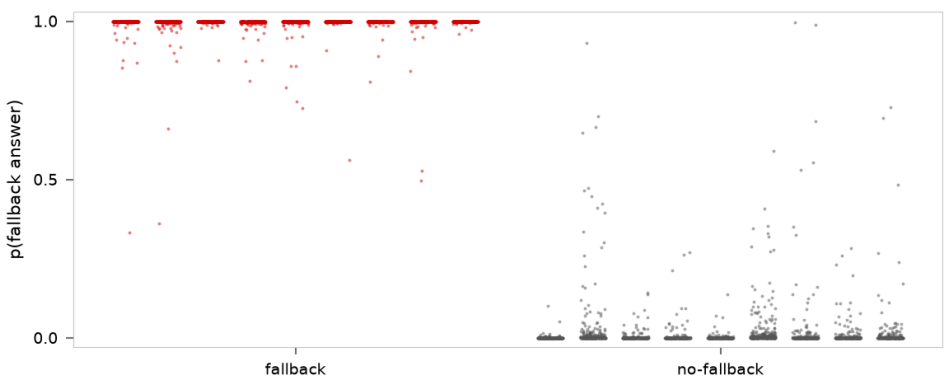
- **`redirect`** — γ = 2 at the embedding, at every position, with no labels. This is the same edit training applied, so H1 reads the readout the term trained. Read by H1 and the first clause of H3.
- **`projection`** — the ex-2.2.1 intervention: γ = 1 at every slice and every position. The fallback was never trained under it. Read by the second clause of H3 and the projection row of H4.
- **The transfer sweep** — γ ∈ {0.5, 1, 1.5, 2} at the embedding, every position. Its γ = 2 point is `redirect`, and its γ = 1 point is the `embedding` arm of ex-2.2.1, which differs from `projection` in leaving the later slices alone. Read by H4.
- **Ride-along rows** — the `operands`, `shaped`, and `ablate` arms from ex-2.2.1, run on the fallback condition without gates. That way the intervention-tuning pass in the design has the fallback rows to hand (E5). `operands` is the one row here that needs position labels; it stays so we can compare with ex-2.2.1.

The designed response (H1)

H1. Under `redirect`, the model's answer on red lines is the fallback answer. Seed-mean fallback accuracy over the red lines with a clean visible operand (296 of the 365) is at least 0.8; partial: between 0.5 and 0.8. The other 69 red lines have no defined fallback answer, so we report them beside the gated figure, unscored. The reference row is the no-fallback condition under the same intervention. The fallback figure has to sit above that reference by a resolved margin, meaning two pooled between-seed standard deviations. A figure that clears 0.8 without a resolved margin counts as partial. Contrary: fallback accuracy at the no-fallback level, which would mean the term did not train the readout. The fallback loss over training (see the method) says whether the term was ever active.

condition	fallback acc., clean visible	true-answer acc., clean visible	fallback acc., red visible	true-answer acc., red visible
un-anchored	0.000 (0.000–0.000)	0.992 (0.990–0.993)	0.00 (0.00–0.00)	1.00 (0.99–1.00)
no-fallback	0.015 (0.000–0.044)	0.034 (0.003–0.088)	0.00 (0.00–0.00)	0.00 (0.00–0.00)
fallback	0.999 (0.997–1.000)	0.000 (0.000–0.000)	0.00 (0.00–0.00)	0.00 (0.00–0.00)
fallback, no anti-anchor	0.997 (0.997–0.997)	0.000 (0.000–0.000)	0.00 (0.00–0.00)	0.00 (0.00–0.00)
anti-anchor, no fallback	0.011 (0.000–0.030)	0.014 (0.003–0.024)	0.00 (0.00–0.00)	0.01 (0.00–0.03)
fallback at w_fb = 0.01	0.998 (0.993–1.000)	0.000 (0.000–0.000)	0.00 (0.00–0.00)	0.00 (0.00–0.00)
fallback at w_fb = 0.25	0.995 (0.993–1.000)	0.000 (0.000–0.000)	0.00 (0.00–0.00)	0.00 (0.00–0.00)
the recipe through the new code path	0.014 (0.000–0.041)	0.018 (0.000–0.034)	0.00 (0.00–0.00)	0.00 (0.00–0.00)

Seed mean with the seed range, under `redirect`. The "clean visible" columns are the 296 lines H1 gates. The "red visible" columns are the other 69 red lines, which have no defined fallback answer; the scorer reads it as the visible operand mixed with gray all the same, so that column says whether the model emits that anyway.



The probability on the fallback answer, per seed. *Under* `redirect`*, each dot is one of the 296 red lines with a clean visible operand; a column is one seed. Left, the fallback condition; right, the no-fallback condition. A split response, with some lines at the fallback answer and some elsewhere, would show as a column with dots at both ends.*

Under `redirect`, the fallback condition emits the fallback answer on 0.999 of the red lines with a clean visible operand, in every seed; the gate is 0.8. Its margin over the no-fallback condition, which emits it on 0.015, clears the 0.025 resolution floor.

The probability the model puts on the fallback answer is 0.996, averaged over seeds and lines, and the strip shows no split: every seed puts nearly every line at the top. The un-anchored condition keeps its true answer under the same edit (0.992), so the reflection does its work only where a concept was placed. On the 69 red lines whose visible operand is itself red, the fallback condition emits neither the true answer nor the gray mix of the visible operand.

H1 holds. The term trained the readout: the fallback loss over training (next section) falls to near zero, and the reflected state decodes to the designed answer on all but a handful of qualifying lines, in every seed.

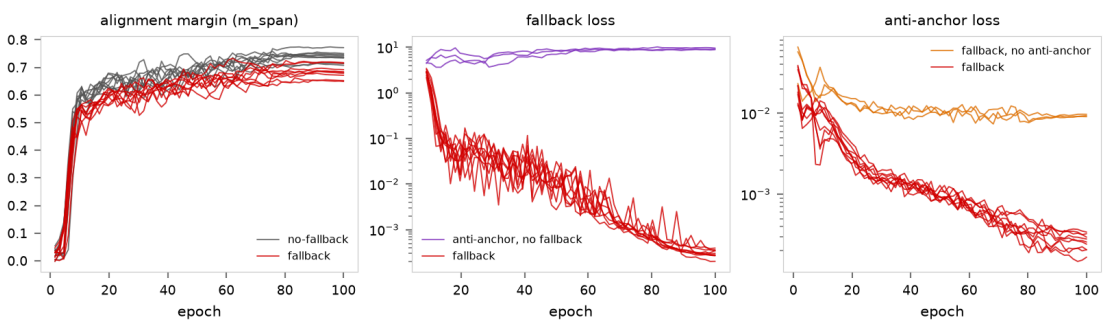
Task and placement intact (H2)

H2. The term costs nothing on the task or the placement. Clean exact-match accuracy on all probe lines is within 0.02 of the no-fallback condition, and the seed-mean alignment margin at the end of training, the `m_span` of ex-2.1.10, is at least 0.8 of the no-fallback value. Partial: exactly one of the two holds, or both hold with accuracy read at the wider 0.05 band.

Contrary: the fallback term competing with the anchor for the operand states, which is what the stop-gradient is supposed to prevent.

condition	clean acc., all	red	non-red	margin `m_span`	mean α at op1
un-anchored	1.000 (0.999–1.000)	0.999 (0.997–1.000)	1.000 (0.999–1.000)	0.028 (0.003–0.070)	-0.041 (-0.072–0.011)
no-fallback	0.999 (0.997–1.000)	0.999 (0.995–1.000)	0.998 (0.996–1.000)	0.738 (0.708–0.772)	0.050 (0.008–0.087)
fallback	0.999 (0.995–1.000)	0.997 (0.984–1.000)	0.998 (0.992–0.999)	0.682 (0.649–0.717)	0.105 (0.079–0.136)
fallback, no anti-anchor	0.999 (0.998–0.999)	1.000 (1.000–1.000)	0.999 (0.998–1.000)	0.736 (0.723–0.749)	0.043 (0.009–0.087)
anti-anchor, no fallback	0.998 (0.996–1.000)	0.994 (0.981–1.000)	0.999 (0.996–1.000)	0.697 (0.678–0.724)	0.106 (0.074–0.127)
fallback at w_fb = 0.01	0.999 (0.996–1.000)	0.996 (0.992–1.000)	1.000 (0.999–1.000)	0.677 (0.661–0.691)	0.111 (0.086–0.138)
fallback at w_fb = 0.25	0.998 (0.994–1.000)	0.995 (0.984–1.000)	0.999 (0.996–1.000)	0.690 (0.670–0.709)	0.101 (0.076–0.138)
the recipe through the new code path	0.999 (0.998–1.000)	0.999 (0.997–1.000)	0.999 (0.998–1.000)	0.738 (0.724–0.760)	0.058 (0.031–0.089)

Seed mean with the seed range on the clean pass. The margin is `m_span` from ex-2.1.10: the pooled alignment contrast between the labelled and unlabelled operand states. The last column is the mean alignment of every op1 state, which is the containment statistic of ex-2.1.10.



Training, seed by seed. *Left, the alignment margin `m_span` over training, one thin line per seed: the fallback condition over the no-fallback condition. Middle, the fallback term's loss, in the fallback condition and in the `anti-only` arm, where it is measured on every step and never trained. Right, the anti-anchor hinge, in the fallback condition and in the `fb-only` arm, where it is measured and not trained. The losses are means over the crops between two trajectory records, and the fallback loss is a mean over the qualifying lines in those crops.*

Clean accuracy on all probe lines is 0.999 in the fallback condition against 0.999 without the term, a gap of 0.0002 against a gate of 0.02. The margin at the end of training is 0.682 against 0.738, a ratio of 0.925 against a gate of 0.8. Every fallback seed sits between 0.649 and 0.717.

The margin is lost to the anti-anchor hinge: the `fb-only` arm ends at 0.736 and the `anti-only` arm at 0.697. The hinge also raises the mean alignment at op1, from 0.050 to 0.105.

The fallback loss switches on once the anchor has placed the concept, inside the warm-up, and reaches 0.0003 by the end; in the `anti-only` arm, where it is measured and never trained, it stays at 9.2. The anti-anchor hinge ends at 0.0003 where it is trained and 0.0094 where it is not.

H2 holds. Both clauses clear their gates: the term costs nothing on the task, and the margin stays above 0.8 of the no-fallback value.

Selectivity (H3)

H3. Under `redirect`, the seed-mean non-red deficit stays at or below 0.02. Under `projection`, the non-red deficit in the fallback condition is not resolved above the no-fallback figure (0.024 in ex-2.2.1, re-measured here).

On a non-red line the operands have nearly nothing on the axis, so the first clause rests on the syntax embeddings. Zeroing their constant component is what cost the ex-2.2.1 projection its 0.024, and the reflection flips it instead. Training runs the reflected pass on every crop that carries a qualifying red line, so the blocks see flipped syntax states, and read them as syntax, throughout training. We predict that this transfers to the non-red lines the term never scored.

Partial: the first clause holds only at the 0.05 band, or exactly one clause holds.

Contrary on the first clause: a deficit at or above the ex-2.2.1 figure, which would say the flipped syntax reads as a different syntax, and would send the intervention-tuning pass toward a thresholded reflection, the `shaped` falloff at strength 2. Contrary on the second clause: the term widens the edit, which would mean the blocks now read the axis at positions or slices where they did not before.

condition	non-red deficit, `redirect`	non-red damage, `redirect`	non-red deficit, `projection`	non-red damage, `projection`
un-anchored	0.022 (0.001–0.046)	0.033 (0.006–0.061)	0.000 (-0.001–0.001)	0.005 (0.003–0.008)
no-fallback	0.774 (0.513–0.961)	0.825 (0.591–0.961)	0.024 (0.001–0.127)	0.034 (0.007–0.158)
fallback	0.749 (0.607–0.889)	0.747 (0.607–0.884)	0.114 (0.035–0.217)	0.138 (0.053–0.246)
fallback, no anti-anchor	0.817 (0.789–0.869)	0.812 (0.790–0.854)	0.168 (0.078–0.229)	0.190 (0.097–0.244)
anti-anchor, no fallback	0.735 (0.621–0.837)	0.812 (0.719–0.901)	0.016 (0.009–0.022)	0.030 (0.022–0.042)
fallback at w_fb = 0.01	0.765 (0.673–0.854)	0.764 (0.679–0.845)	0.047 (0.011–0.104)	0.068 (0.025–0.135)
fallback at w_fb = 0.25	0.775 (0.607–0.944)	0.770 (0.610–0.926)	0.243 (0.057–0.602)	0.256 (0.079–0.598)
the recipe through the new code path	0.910 (0.866–0.992)	0.943 (0.915–0.983)	0.023 (0.004–0.035)	0.034 (0.008–0.051)

Seed mean with the seed range. The deficit is the drop in exact-match accuracy on the 1689 non-red lines from the clean pass; damage is the drop in probability on the true answer.

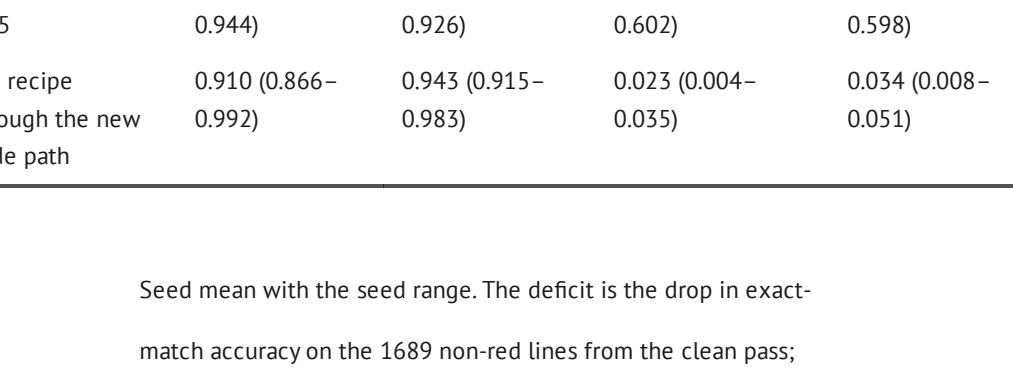
First clause. Under `redirect`, the fallback condition loses 0.749 of its non-red accuracy, with no seed under 0.607; the gate is 0.02 and the partial band 0.05. The reflection is no more selective without the term: the no-fallback condition loses 0.774 under it, the `recipe` arm 0.910, and the un-anchored condition 0.022.

So a full reflection of every embedding state is destructive on non-red lines in every anchored model, and the training-time exposure to flipped syntax states did not transfer to them. What those lines decode to is in the post-hoc row of the exploratory section.

Second clause. Under `projection`, the non-red deficit in the fallback condition is 0.114 against 0.024 without the term, which is what ex-2.2.1 reported on the same checkpoints. The difference, 0.090, sits under the resolution floor of 0.124, so by the frozen rule the deficit is not resolved above the reference.

The floor is wide because both conditions spread over seeds: the fallback seeds run from 0.035 to 0.217, the reference seeds from 0.001 to 0.127. Damage tells the same story (0.138 against 0.034, floor 0.133).

The weight bracket is the sharper reading: the deficit under `projection` rises with the weight, from 0.047 at 0.01 through 0.114 at 0.05 to 0.243 at 0.25.



The bound and the write under `projection`, fallback condition. *The 99th-percentile alignment over the non-red lines at each (slice, position), seed mean, on the shared 0–1 scale. Left, the clean map, whose arcsine is the bound; right, the alignment arriving at the operator under `projection`, whose arcsine is the write. Slices run from the embedding (lightest) to the last block (darkest). A deeper line above its clean value would mean a block re-writes the axis after it was removed upstream, which is what a wider edit looks like.*

The write map shows no re-writing at any site: every deeper slice arrives at the operator at or below its clean alignment. Whatever the term changed under `projection`, it did not widen where the edit acts on the axis.

H3 is partial, by the frozen rule that exactly one clause holds. The first clause fails, and by more than the partial band allows. That is the contrary case named above: the flipped syntax does not read as syntax. The second clause holds, but as an unresolved difference, and the weight bracket suggests the difference is real and grows with the weight.

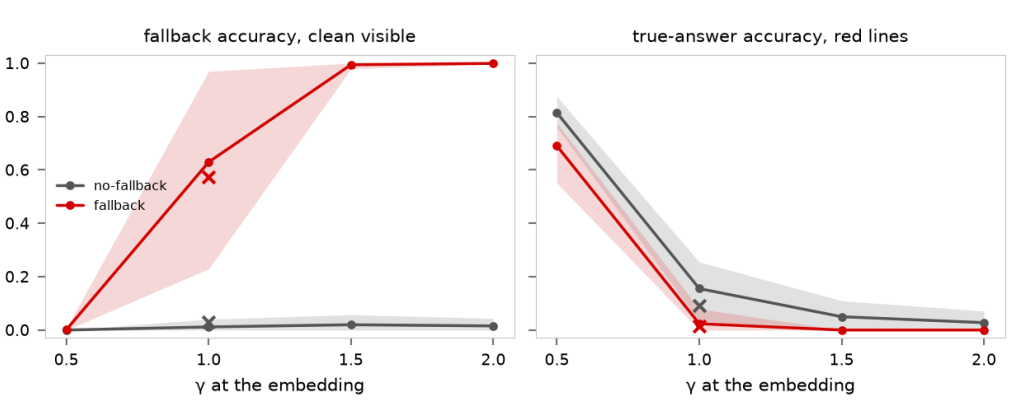
Transfer from the antipode to zero (H4)

H4. The designed response transfers (generalizes) part of the way to zero. Along the transfer sweep, $\gamma \in \{0.5, 1, 1.5, 2\}$ at the embedding, seed-mean fallback accuracy on red lines is non-decreasing in γ, allowing a dip of at most 0.02 between adjacent strengths, and at $\gamma = 1$ it is at least half its value at $\gamma = 2$. Partial: either clause holds on its own — monotone with $\gamma = 1$ below half, or $\gamma = 1$ at half or more with a dip larger than the allowance.

The $\gamma = 1$ point is a real removal, not an inert midpoint. It is the `embedding` arm of ex-2.2.1, where zeroing the axis at the embedding alone took red accuracy down to 0.16. The sweep adds how the fallback responds along the way from the trained state to that one. At $\gamma = 1$ the operator keeps whatever the state carries off the axis and rescales it by $1/\sqrt{1-\alpha^2}$, so the landing direction is defined as long as the state is not perfectly aligned, and no state is: pure red arrives at the embedding at about $\alpha = 0.9$, a gain of about 2.3.

Contrary: fallback accuracy at $\gamma = 1$ sits at the no-fallback level and the rise is confined to $\gamma > 1$. That would be the mismatch showing in full, with the response living at the antipode and not reaching the projected state. The concept swap filed for D2.3 would address it by targeting a state training already visits; widening the bracket here would not.

We report the `projection` row beside the sweep, without a gate: $\gamma = 1$ at every slice rather than at the embedding alone. The difference between the two says how much the projection at later slices costs the designed response.



The transfer sweep. *Seed mean with the seed range as a band, for the projection operator at the embedding at strength γ, every position. Left, fallback accuracy on the red lines with a clean visible operand; right, true-answer accuracy on all red lines. $\gamma = 2$ is `redirect`; $\gamma = 1$ is the `embedding` arm of ex-2.2.1. The crosses at $\gamma = 1$ are the `projection` row of each condition, which removes the axis at every slice rather than at the embedding alone.*

Along the sweep, the seed-mean fallback accuracy in the fallback condition runs 0.001, 0.630, 0.994, 0.999 at $\gamma = 0.5, 1, 1.5, 2$. It is non-decreasing: the largest drop between adjacent strengths is 0.000, against an allowance of 0.02. At $\gamma = 1$ it stands at 0.63 of its value at $\gamma = 2$, against a gate of 0.5.

The seed range at $\gamma = 1$ is wide, from 0.23 to 0.97, and at $\gamma = 1.5$ every seed is above 0.98. The no-fallback condition stays at 0.020 or below throughout. True-answer accuracy on red lines falls the same way in both conditions: the removal at $\gamma = 1$ is at least as complete with the term as without it (0.024 against 0.155).

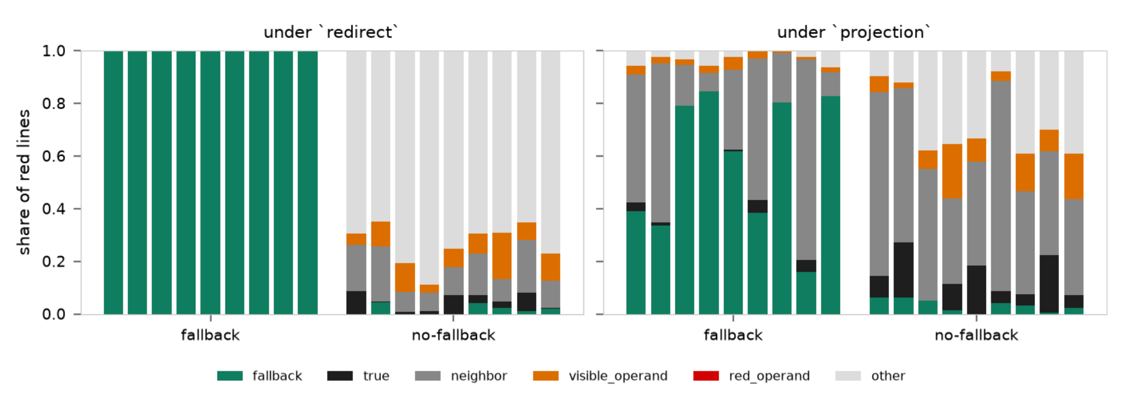
The `projection` row, $\gamma = 1$ at every slice, gives 0.572 (0.16 to 0.84), a little under the embedding-only figure at the same strength. Removing the axis at the later slices as well costs the designed response 0.057 on the seed mean.

H4 holds. The response reaches the projected state at more than half strength, and rises monotonically from there to the trained one. The contrary case, a rise confined to $\gamma > 1$, did not occur.

Exploratory analyses

Preregistered as exploratory, no gates, except the last two rows, which are post hoc.

E1 — composition.



What red lines decode to. *Each bar splits the 296 red lines with a clean visible operand by the decoded answer, per seed: the fallback answer, the true mix, a one-step neighbor of it, the visible operand, the red operand, or something else, tested in that order. Left, under `redirect`; right, under `projection`. The nine fallback seeds sit left of the nine no-fallback seeds in each panel.*

Under `redirect`, the fallback condition decodes 0.999 of red lines to the fallback answer, and puts 0.000 of its mass outside the color vocabulary. The no-fallback condition puts 0.228 outside it under the same edit, with a seed range from 0.01 to 0.67. Under `projection` the fallback condition splits between the fallback answer and a one-step neighbor of the true mix, in a proportion that varies by seed.

condition	agree, `redirect`	plurality, `redirect`	agree, `projection`	plurality, `projection`
un-anchored	–	0.992	–	0.999
no-fallback	0.024	0.260	0.142	0.358
fallback	1.000	0.999	0.726	0.592
fallback, no anti-anchor	–	0.997	–	0.541
anti-anchor, no fallback	–	0.474	–	0.481
fallback at w_fb = 0.01	–	0.998	–	0.503
fallback at w_fb = 0.25	–	0.997	–	0.839
the recipe through the new code path	–	0.439	–	0.468

Seed agreement on the red lines with a clean visible operand. "Agree" is the fraction of lines on which at least 5 of the nine seeds decode the same answer, the statistic from ex-2.2.1, which put it at 13% under the projection; it is undefined for the three-seed conditions. "Plurality" is the mean over lines of the fraction of seeds decoding the line's plurality answer, defined for every condition.

E2 — the antipode. The fraction of clean states with negative alignment, per slice and averaged over positions:

condition	emb.	block 1	block 2	block 3	block 4	all
un-anchored	0.569	0.608	0.632	0.691	0.625	0.625
no-fallback	0.374	0.281	0.263	0.183	0.310	0.282
fallback	0.065	0.110	0.065	0.015	0.034	0.058
fallback, no anti-anchor	0.377	0.249	0.260	0.224	0.351	0.292
anti-anchor, no fallback	0.051	0.101	0.073	0.023	0.044	0.058
fallback at w_fb = 0.01	0.051	0.092	0.046	0.013	0.038	0.048
fallback at w_fb = 0.25	0.065	0.092	0.079	0.022	0.042	0.060
the recipe through the new code path	0.366	0.231	0.232	0.156	0.253	0.248

The anti-subspace term alone leaves about a quarter of clean states past the antipode plane; the hinge takes that to about 0.06, and the `fb-only` arm, without the hinge, sits where the no-fallback condition does.

E3 — off-axis recoverability. Held-out R^2 of a ridge probe for the concept operand's redness, fitted on the operand states with the axis deleted, per slice. (A ridge probe is a linear regression with a penalty on the weights; held-out R^2 is the share of variance it explains on lines it was not fitted to.) Fallback / no-fallback, with the resolution floor (two pooled between-seed sds) beside each:

states	emb.	block 1	block 2	block 3	block 4	floor: emb.	b1	b2	b3	b4
clean	0.915 / 0.896	0.930 / 0.932	0.929 / 0.927	0.922 / 0.918	0.902 / 0.896	0.045	0.024	0.030	0.034	0.045
redirect	0.915 / 0.896	0.897 / 0.866	0.891 / 0.849	0.879 / 0.839	0.854 / 0.812	0.045	0.078	0.054	0.057	0.055
projection	0.952 / 0.949	0.940 / 0.931	0.928 / 0.904	0.906 / 0.872	0.876 / 0.830	0.020	0.033	0.040	0.052	0.075

The floor the task itself sets, the same ridge fit to the raw RGB values, is 0.865. Every clean figure is above it; under the interventions the deeper slices dip below it, as far as 0.812 at the last block. Under `redirect` and under `projection`, the R^2 in the fallback condition sits above the no-fallback one at the deeper slices, by 0.03 to 0.05; that is under the floor at every slice. The follow-up at more seeds is where that would be resolved.

E4 — arms. The tables under H1, H2, and H3 carry every arm. Dropping the hinge (`fb-only`) leaves the fallback under `redirect` intact and the margin at the no-fallback level, and takes the transfer to $\gamma = 1$ from 0.630 to 0.399. Dropping the fallback term (`anti-only`) leaves no designed response.

Along the weight bracket, transfer to $\gamma = 1$ rises with the weight (0.354, 0.630, 0.775) and so does the non-red deficit under `projection` (see the table under H3).

The `recipe` arm reproduces the stored no-fallback checkpoints through the new code path: margin 0.738 against 0.738, red accuracy under `projection` 0.120 against 0.091, non-red deficit under `projection` 0.023 against 0.024. The recomputed clean alignment maps of the stored runs match the published ones from ex-2.1.10 to 0.0011.

E5 — ride-along interventions.

intervention	condition	red acc.	fallback acc., clean visible	non-red deficit	non-red damage
`operands`	fallback	0.064 (0.000–0.184)	0.138 (0.017–0.270)	0.000 (-0.001–0.001)	0.000 (-0.001–0.001)
`operands`	no-fallback	0.125 (0.000–0.263)	0.030 (0.000–0.068)	0.000 (-0.001–0.002)	0.001 (0.000–0.005)
`shaped`	fallback	0.436 (0.203–0.775)	0.015 (0.000–0.051)	0.000 (0.000–0.000)	-0.000 (-0.000–0.000)
`shaped`	no-fallback	0.600 (0.260–0.841)	0.003 (0.000–0.017)	0.000 (0.000–0.000)	-0.000 (-0.000–0.000)
`ablate`	fallback	0.019 (0.000–0.074)	0.491 (0.081–0.801)	0.104 (0.024–0.208)	0.126 (0.044–0.233)
`ablate`	no-fallback	0.119 (0.000–0.249)	0.020 (0.000–0.041)	0.012 (0.001–0.044)	0.020 (0.003–0.065)

Seed mean with the seed range, for the three ex-2.2.1 edits run without gates on both nine-seed conditions. Red accuracy and fallback accuracy are read on red lines, the deficit and damage on non-red lines.

The `operands` edit, at the operand positions only, keeps its selectivity in the fallback condition and produces the fallback answer on 0.14 of lines. `ablate` is the one row where the fallback condition differs from the reference by a large amount: it emits the fallback answer on about half the red lines and loses 0.104 of its non-red accuracy, against 0.012 without the term.

E6 — a LUNAR-style redirect, fitted after training. On each frozen no-fallback checkpoint we fit one 64×64 matrix at the embedding, applied at every position, on the qualifying red lines the fallback term uses. The loss has two parts: the fallback cross-entropy at `=`, and an identity term on every other state, meaning the mean squared distance from the unedited state.

This model has no refusal region of its own, so the fallback answer stands in for it. The fitted matrix runs through the eval contract like any other operator, giving fallback accuracy 0.989 (0.966–1.000) on the red lines with a clean visible operand, red-line accuracy 0.000 (0.000–0.000), a non-red deficit of 0.388 (0.144–0.852), and non-red damage of 0.408 (0.161–0.847).

Beside the fallback condition under `redirect` (0.999 and a deficit of 0.749), the fitted edit reaches the same designed response at a smaller cost on non-red lines, with a wide seed spread. Its identity term was the only thing asking it to leave other positions alone, and at the weight used it did not do so.

E7 — non-red lines under the reflection (post hoc). What the non-red lines decode to, as a fraction of (seed, line) pairs: the true answer, the fallback answer of the visible operand, the visible operand itself, or something else.

condition	true, `redirect`	fallback	visible	other	true, `projection`	fallback	visible	other
un-anchored	0.978	0.003	0.001	0.018	0.999	0.000	0.000	0.001
no-fallback	0.224	0.007	0.021	0.748	0.975	0.001	0.000	0.024
fallback	0.249	0.226	0.001	0.524	0.884	0.028	0.001	0.087
fallback, no anti-anchor	0.182	0.202	0.001	0.615	0.831	0.037	0.000	0.132
anti-anchor, no fallback	0.264	0.003	0.046	0.687	0.982	0.000	0.002	0.015
fallback at w_fb = 0.01	0.234	0.186	0.000	0.579	0.952	0.011	0.001	0.036
fallback at w_fb = 0.25	0.224	0.183	0.000	0.593	0.756	0.072	0.000	0.172
the recipe through the new code path	0.088	0.009	0.010	0.893	0.976	0.001	0.002	0.021

Under `redirect`, the fallback condition emits the gray mix of the visible operand on 0.23 of non-red (seed, line) pairs, so about a third of its loss on those lines is the designed response firing where no concept was removed. The rest is undesigned, as is the whole of the loss in the no-fallback condition. Under `projection` the fallback fires on 0.028 of them, which accounts for about a third of the unresolved difference in the second clause of H3.

E8 — the syntax embeddings on the axis (post hoc). 99th-percentile clean alignment over the non-red lines at the embedding slice, one figure per position, averaged over seeds. Operand and answer columns pool all the color tokens, so their percentile is the tail of about two hundred tokens; a syntax column has only one token, so its percentile is that token's alignment.

condition	`op1`	`+`	`op2`	`=`	`ans`	`⏎`
un-anchored	0.31	0.05	0.31	0.04	0.31	0.08
no-fallback	0.17	0.28	0.17	0.40	0.17	0.05
fallback	0.09	0.29	0.09	0.33	0.08	0.06

In the un-anchored model the `+` and `=` embeddings sit where a random direction would. In every anchored model they carry 0.3 to 0.4. Meanwhile the anti-subspace term has flattened the non-red colors, and the hinge flattens them further. So the leak sits in two token embeddings at slice 0, before any block runs.

Discussion

The fallback in the transformer does what it did in M1. At the edit it was trained at, every seed gives the designed answer, and the seed disagreement ex-2.2.1 found under projection is gone there.

What is new is the gap between the edit the term was trained at and the edit we want to use. The clean model never visits the antipode, and the response trained there carries over to the plain projection in most seeds, but in some barely at all. That is the mismatch the design named, and the anchored-op experiments should train toward a state that training already visits; the concept swap filed for D2.3 gives one by construction.

The anti-anchor hinge pays for that transfer with margin: it keeps the clean states on the anchor side of the plane. That cost comes from designing at the antipode, and the swap has no antipode, so we are not adopting the hinge into the recipe. Nor does the fallback term itself carry into the operator experiments by default. We are keeping the code, for fits like E6 and for a concept with no visited state to aim at.

The selectivity cost belongs to the edit rather than to the term. Every anchored model loses most non-red lines under the reflection, because the `+` and `=` embeddings carry the axis (E8) and the reflection flips them along with the operand. We read the tied readout as what puts the axis there: the state after a red operand sits on the axis, and the cheapest way to predict the token that follows is for that token's readout to lean the same way. An untied readout would test this.

Until those embeddings are clean, any edit applied at every position pays the cost, including a rotation to a second anchored concept. The thresholded and operand-only edits of ex-2.2.1 avoid it.

On masking, the off-axis row cannot say whether the term keeps red readable elsewhere. The shift it shows is inside the floor, and the follow-up at more seeds is filed.

The fitted edit of E6, which reaches the designed response at a smaller non-red cost, is the LUNAR argument in miniature: if the model already produces the destination, there is no need to train at a state it never visits. The trained term still has one thing going for it, the response living in the model rather than in a fitted map. The fit of E6 could be tightened before the two are compared again.

Method

Training

We use the primary recipe from ex-2.1.10 unchanged: the `v216` corpus, d64-L4, 100 epochs with a 10-epoch warm-up, the pooled either-operand labeller at τ = 0.1, the anchor at λ = 0.1 with its anneal, and the anti-subspace term at the ex-2.1.8 operating point. The ex-2.1.11 survey proposed a different point. But survey numbers are proposals until they are re-measured at fresh seeds, and the design schedules that confirmation on the multi-op grammar. Keeping the ex-2.1.10 recipe also keeps its stored checkpoints as the no-fallback condition, so nothing has to be retrained for the comparison.

We also use the same seeds as the primary of that experiment, so the fallback and no-fallback conditions differ only in the two new terms. Both are weights on the existing anchored train step.

The fallback term. On each training crop, we reflect the embedding state of every position through the axis, $h \mapsto \mathrm{normalize}(h - 2\alpha\, e_1)$, and hold the reflected states fixed with a stop-gradient. We then run the blocks forward from there and take the cross-entropy at the `=` position of each qualifying line against its fallback token. A line qualifies on three counts: its dose is at least 0.8, the redness of its visible operand is below 0.5, and the clean embedding alignment of its concept operand is at least 0.5. The reflection singles out no positions, so it is the same edit `redirect` applies at eval, and it applies unchanged to a model whose positions are not labelled.

Compute. The reflected pass is a second forward and backward through the blocks, so a step that carries a qualifying line costs about twice a plain step. A crop holds about ten lines, and about one line in twenty qualifies, so roughly 40% of crops carry one. Run per batch, that is nearly every step, and the budget below counts the term as a doubling. Three things should make it cheap at scale. The pass can be skipped when no position clears the alignment threshold, which the clean pass at the edit slice already reports, so it costs nothing before the anchor has placed the concept. It can run on a subsample of the qualifying sequences, or on every k-th step, since the term is a small weight on a slowly moving target. And it reruns only the part of the model after the edit, so an edit deeper than the embedding shares the earlier layers with the clean pass.

The term is the mean over qualifying lines, at a constant weight of 0.05 from step 0. We take that weight from M1, as a starting point rather than a derived value: there it scaled a mean squared error on one decoder, and here it scales a cross-entropy through four blocks. Checking it is what the bracket arms are for.

The alignment threshold keeps the term inert until the anchor has placed the concept, which the ex-2.1.10 trajectories put inside the warm-up. If the concept never arrived at the edit slice, the term would stay inert rather than train toward the wrong state, and the margin gate of H2 would report the anchoring failure.

Where the term acts. The stop-gradient makes this similar to the decoder-only term from M1: the reflected embedding states are detached, so nothing flows back through them to the placement of *red* at the edit, while the four blocks and the readout do learn. The readout is the embedding table itself, since nGPT ties the two, so the table sees the term through the readout only: the answer tokens' readout vectors move toward the reflected pass's final state, and every other token's readout, *red*'s included, moves a little away from it. That is a decoder-side nudge on the readout table rather than a pull on where the concept sits, and the margin gate of H2 is where it would show. Together the blocks and the readout learn a map from a state at the antipode to the fallback answer. In ex-2.2.1 the concept was read from the operand states in the first two blocks, so those are the blocks with something to learn.

We reflect once, at the first anchored slice, which in this recipe is the embedding. Reflecting at every slice, as the ex-2.2.1 projection does, would need a detach at every slice, and then only the layers after the last detach would train; reflecting everywhere and detaching only the first edit is the rehearsal the design rejected.

The stop-gradient does leave one thing open. The blocks after the edit are shared with the clean pass, so the term can reshape how the concept is carried downstream of the edit, including keeping it readable off the axis. That is what the off-axis row (E3) and the margin gate watch.

The anti-anchor term. This is $\mathrm{mean}(\max(-\alpha, 0))$ over the same live positions and slices the anti-subspace term reads, on the clean forward pass only, at a constant weight of 0.1. The reflected states that the fallback term produces are not live positions of that pass, so the two terms do not pull against each other. The term is a hinge on negative alignment: zero when the alignment is positive, and growing linearly as it goes negative. So it keeps clean states out of the half of the space where the fallback lives, and because it acts on one side only, it does not oppose the anchor. The anti-subspace term already penalizes α^2 there, so the arms that drop one term or the other say whether the hinge adds anything.

The fallback answer

Removing the concept operand leaves the visible operand and an unknown partner. The least committal answer is the distribution of the mix over every partner the corpus allows, which we call the *operand-averaged null*. On this grid, each channel of the visible operand has three closed partners, and their three mixes are distinct, so the null is uniform over 27 colors and has no mode.

Its per-channel median is the mix with the middle partner. That coincides, at every level, with the visible operand mixed with mid-gray (7.5 on the 16-level channel) and rounded to the nearest grid level; `experiment.py` asserts the coincidence. That color is the fallback answer, and it is the gray fallback from M1 carried over.

The fallback answer depends on the visible operand alone, so it is defined only when that operand is clean. On a red line whose visible operand is itself red (redness ≥ 0.5), both operand states are reflected. The answer would then be the mix with both operands replaced by mid-gray, which is 7.5 on every channel, halfway between grid levels 6 and 9. With nothing visible, the null has no center on the grid; when one operand is visible, its level always breaks the tie. Those 69 lines take no fallback loss and sit outside the gate of H1.

Measurements

We run one teacher-forced pass per run per intervention, over all 5,832 lines, through the eval contract in `sca.intervention`. Each pass gives the log-softmax at the `=` position, from which we read the probability on the correct answer and on the fallback answer, the argmax, and the mass outside the color vocabulary.

Each pass also gives the write per (slice, line, position), as in ex-2.2.1, and the clean alignment map. Fallback accuracy, seed agreement, and the composition all come from the argmax on red lines. The training trajectory records the fallback and anti-anchor losses beside the anchor loss and the margin, every 50 steps.

Noise floor. As in ex-2.2.1: for each group statistic we take the pooled between-seed standard deviation per condition, computed before any comparison is read. A difference between conditions or arms smaller than twice that value is reported as not resolved. Gates score seed means against fixed thresholds and do not use the floor.

Budget

Twenty-four training runs of the ex-2.1.10 recipe, each at about twice the cost of a plain run (see Compute), plus the scoring of twelve stored checkpoints. The scoring is CPU work, at about a second per run per intervention. E6 adds nine fits of a 64×64 matrix, one per no-fallback seed, at a few CPU minutes each.

---

1. An identity in the forward pass with a gradient of zero (`.detach()` in PyTorch), so a loss downstream of it cannot move anything upstream of it. The other losses see the clean pass and are unaffected. ↩

2. A KL term penalizes the distance between the next-token distribution of the model and that of the reference. ↩