Ex 2.1.2: making composition necessary

> The previous model never answered a held-out *named* pair, so we changed the grammar to make composition the only route: reverse alias lines, and named equations whose mix falls off the palette. The model picks up both new skills and still won't chain them within one forward pass, leaving held-out named accuracy at zero.

Ex-2.1.1 gave us a decent baseline, but no model in the sweep ever solved `named_holdout`. That eval set contains pairs of named colors whose named-answer equation never appears in training, so the expected way to answer is to combine two skills: the alias dictionary, which says which name goes with which hex color, and the mixing arithmetic itself.

That report looked into why the models never learn to combine them. In this experiment, we keep everything from ex-2.1.1 fixed, including the d64-L4 architecture, the split, and the training recipe, and add two new kinds of sequence:

- Reverse alias lines (`#f00 = red`). The base grammar never trained the hex → name readout.
- Named equations that mix to a color that falls *off* the palette, answered in hex (`red + navy = #804`). Now the surface form of the answer depends on the mix rather than on the operand forms alone, so a name + name prompt can't be settled by lookup.

Type	Example
Named pairs	`red + blue = purple`
Hex pairs	`#f00 + #00f = #808`
Cross-form	`red + #00f = #808`
Alias	`red = #f00`
Reverse	`#f00 = red`
Open	`red + orange = #f40`

We run the two grammar changes as a 2 × 2 factorial. Each one replaces the same number of tokens from the hex slice, so the four conditions differ only in what they add on top of the baseline.

This is still an **un-anchored** experiment.

Training data

We regenerate the `both` corpus here. It shows the two new forms in their natural context: reverse aliases and off-palette named equations, sitting among the lines we already had.

```
#044 + mint = #4a6
#18a + #c0b = #74b
#1bc + #3e5 = #2d9
#0c1 + #890 = #4b1
lavender = #88f
purple + orchid = #c4c
#720 + white = #b98
#f7b + cyan = #8bd
azure + azure = azure
mint + lavender = #8cc
teal = #088
#eba + #bd1 = #dc6
```

condition	hex	named	cross	alias	alias_rev	open
control	23924	6078	6101	3897	—	—
rev	21949	6031	6148	3869	2003	—
open	19921	5964	6153	4065	—	3897
both	17858	6066	6078	4126	1975	3897

Lines per form in the 40,000-line corpus of each condition. The named equations draw from the same 66 training pairs as ex-2.1.1, with the same 10 pairs held out. The open equations draw from 242 of the 302 off-palette named pairs, and 60 are held out entirely.

Hypotheses

Written before looking at any results.

H1. Accuracy: `named_holdout` stays at zero in `control` and is lifted well off zero by `both`. `rev` alone lifts `named_holdout` part of the way: the garden path showed the mix is already half-computed on named prompts, and `rev` supplies the missing readout for whatever computation is there. `open` alone forces the computation, but leaves name emission supervised only through the memorizable named slice.

H2. Margins: scoring all 27 names as complete answers gives each held-out pair a *margin*, the log-probability of the true name minus that of the best competitor. In `control` the margins are negative (accuracy is zero) but spread out well above the random floor. The pairs with the least-negative control margins should be the first to flip positive under intervention. The interventions shift the whole margin distribution upward, while `named_seen` margins stay large and positive.

H3. The answer schedule: probing each answer position for each RGB channel (on hex prompts, strictly before each digit lands in the context) shows a stair-step: channel k stays low and becomes strongly decodable at the position that emits digit k.

H4. Computed but not emitted: a result-color probe fit on open-pair prompts (name + name surface form) at the pre-answer position transfers to the held-out named prompts in proportion to how much the mix is actually computed there: middling R^2 in `control` (partial computation), rising toward the fit ceiling in the `open` conditions, and tracking `named_holdout` accuracy across conditions.

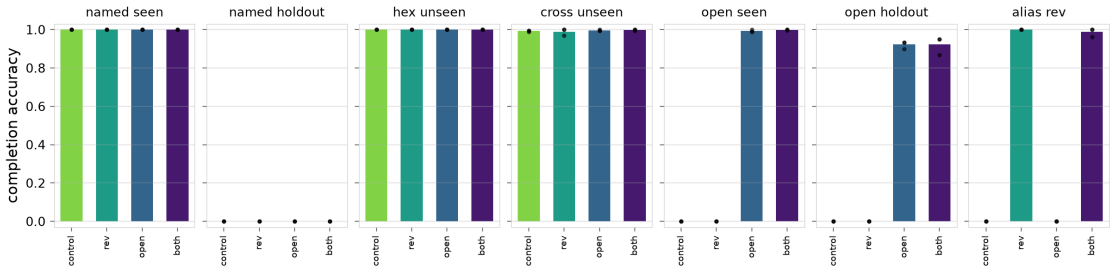
H5. No side effects: `named_seen`, `hex_unseen`, and `cross_unseen` stay saturated in every condition, even though we are displacing up to 15% of the hex data. The `alias_rev` eval set reads ≈ 1 wherever reverse aliases are trained and stays random elsewhere, as in ex-2.1.1.

If the H1 interaction shows up but the single interventions do nothing, then composition needed both ingredients at once. If `rev` alone saturates the set, then the readout was the only missing piece and the pull toward lookup never mattered. If nothing moves, then the diagnosis was wrong somewhere.

Headline numbers. Mean `named_holdout` accuracy by condition: `control` **0.00**, `rev` **0.00**, `open` **0.00**, `both` **0.00**. The saturated sets (named seen, hex and cross unseen) average 0.997, 0.997, 0.999, 0.999 in the same order.

Completion accuracy across the factorial

Did the changes move `named_holdout` off zero, and did the two new forms train the way we meant them to? The figure below shows exact-match accuracy for all seven eval sets, in each of the four conditions.



Each panel is one eval set: the bar is the mean over three seeds and the dots are the individual seeds. `named_holdout` *is the set H1 is about;* `open_holdout` *asks whether the forced computation carries over to off-palette pairs never seen in training;* `alias_rev` *checks the reverse-alias supervision. In* `control` *and* `rev`*, the open-form sets ask for a surface form those corpora never train on (a name + name prompt), so a low score there means the grammar is simply missing from that corpus, rather than a genuine attempt that fell short.*

H1 is refuted: `named_holdout` stays at zero in every condition (its interaction term is a degenerate +0.00). The other panels show that both changes did train, though. The `open` conditions answer held-out off-palette pairs at ≈ 0.9, so the mixing arithmetic runs on name + name prompts and carries over to operand pairs never seen in that surface form. And `alias_rev` reads 1.0 right where reverse aliases were trained, so the hex → name readout exists and works in the frame it was taught in. We supplied both of the missing ingredients, and still the composition does not surface as a named answer.

Right value, wrong spelling

Let's look at what the model actually writes. Below are the greedy completions of the held-out named prompts, meaning we take the single most likely character at each step and read off the answer. In `control` and `rev` the answers are the familiar retrieval-like wrong names. But in the `open` conditions, some of the pairs the model writes a *hex* answer, and when it does, the value is the correct mix.

prompt	expected	control	rev	open	both
`blue + black =`	navy	black	black	black	black
`lime + black =`	green	gray	gray	teal	teal
`black + red =`	maroon	purple	purple	#800 ✓	#800 ✓
`rose + navy =`	purple	gray	navy	#808 ✓	#808 ✓
`blue + cyan =`	azure	teal	teal	black	teal
`blue + magenta =`	violet	lavender	purple	purple	purple
`orchid + azure =`	lavender	gray	gray	#88f ✓	#88f ✓
`white + lime =`	mint	white	white	lime	spring
`chartreuse + maroon =`	olive	maroon	gray	gray	#880 ✓
`olive + lavender =`	gray	lavender	lavender	#888 ✓	#888 ✓

Greedy completions of the `named_holdout` prompts, seed 0, one column per condition. A ✓ marks an answer whose value equals the true mix, but is given in the wrong form.

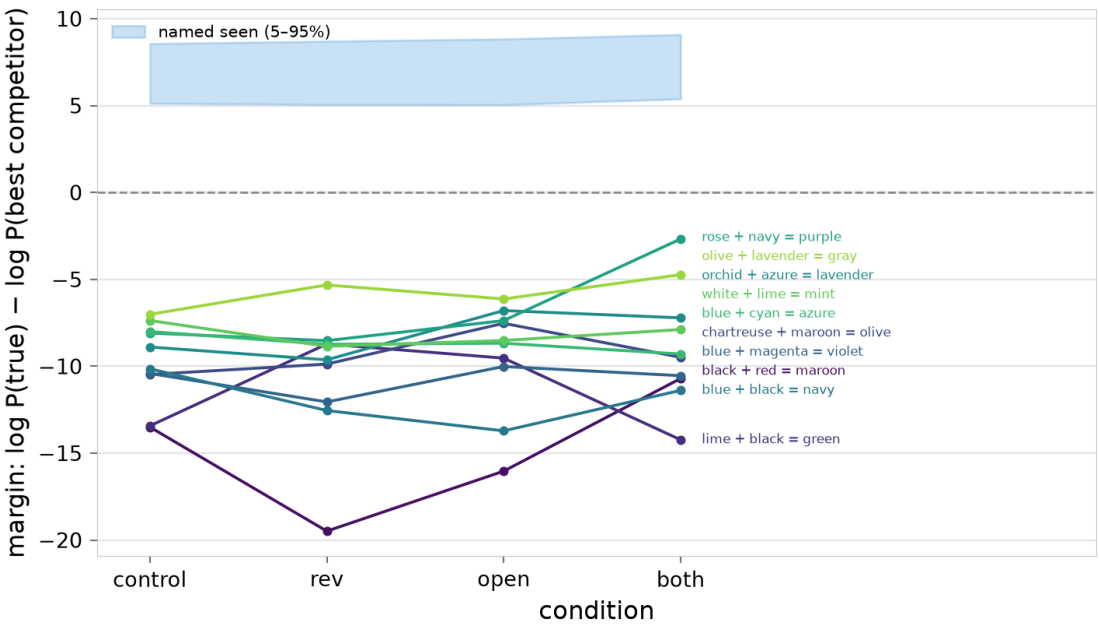
Across all seeds, the hex-form answers on the 10 held-out prompts number `control` **0/30**, `rev` **0/30**, `open` **11/30**, `both` **10/30**, and the value-correct answers `control` **0/30**, `rev` **0/30**, `open` **11/30**, `both` **10/30**. Every value-correct answer is hex, and every name answer has the wrong value. The form-choice margin agrees at `control` -5.2, `rev` -3.3, `open` +8.1, `both` +7.6 nats[1].

So the `open` intervention worked, in that the mix is computed on name + name prompts. But the *form rule* ("answer with a name exactly when the mix lands on the palette") did not carry over: the model treats a held-out closed pair like an open one and answers in hex, or else falls back on a name near the mix. Whether that name is the answer of the closest memorized equation is a further claim we haven't tested here.

So the reverse mapping `rev` didn't help.

Error margins

Exact-match accuracy tells us whether the true name came out on top, but how close it was. So let's grade the name-identity question on a continuous scale.



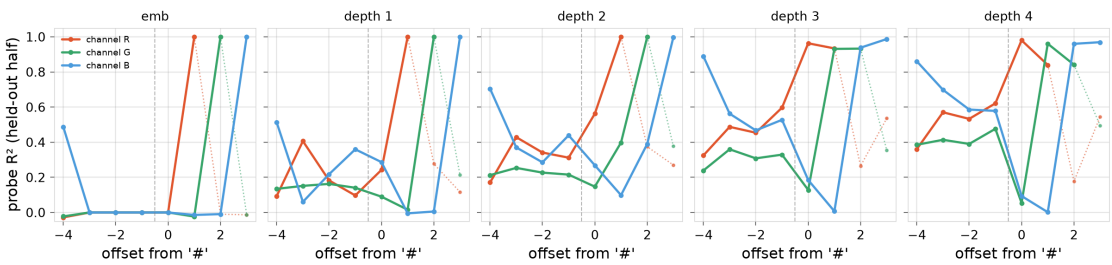
Margins between predicted and true named colors. Positive means the true name comes out ahead of the other names, and the magnitude says by how much. One line per held-out pair, averaged over seeds and traced across the four conditions; the shaded band is the range of `named_seen` margins, for reference.

H2 mostly refuted. The distribution barely moves: the mean margin is -9.7 nats in `control` and -8.8 in `both`, with 0/10 pairs ending positive. The true name never gets close among the names; it stays about ten nats off while `named_seen` is far above zero. So the value → name translation is not *almost there* and narrowly behind. It simply never engages, even in the condition whose greedy answers prove the value has been computed. There is some rank structure, but it is weak: the rank correlation between `control` and `both` is 0.76.

When each channel becomes readable in-sequence

The ex-2.1.1 probes read one pre-answer position and averaged over R, G, and B, which hides at which token the mix gets computed. So here we fit a separate ridge probe for every combination of position, channel, and layer on the hex prompts. We then watch each channel become readable as the answer is spelled out.

H3 predicts a stair-step: each channel stays low until the position that emits its own digit, then climbs sharply.



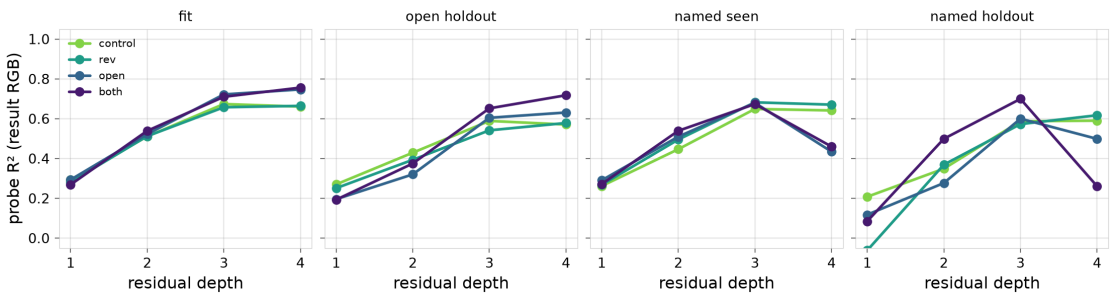
Probe alignment per color channel of the answer. One panel per transformer layer. Offset 0 is the `#`, digit k sits at offset k + 1 and is emitted from offset k. A line is solid where its digit is not yet in the context (so decoding it is computation) and dotted once it has landed (decoding is copying).

At the final layer, channel k is almost perfectly decodable right at its emission offset ($R^2 \approx 0.97$), and *only* there. One position earlier it is much weaker, and once emission moves on to the next digit, the earlier channels mostly fade from the deep residual stream.

So each answer position holds just the one channel it is about to emit, riding on a diffuse trace of the whole mix (around 0.5 R^2) that lingers from the pre-answer position. The mix is never fully represented at any single position; the "result" that the pre-answer probe reads is more of a head start than a finished value. This looks like just-in-time computation.

Is the mix computed on the named prompts?

If the mix really is computed on the named prompts, then a probe trained to read the result somewhere else should carry over to them. So we fit the probe on the open-pair prompts (the name + name surface form) at the pre-answer position, then score it on the named eval sets. Where the mix is never represented, no probe fit elsewhere can recover it.



One panel per scored set: the held-back half of the fit set, open holdout, named seen, and named holdout. R^2 against residual depth, one line per condition (darker means a richer corpus). Depth 0 is left out, since the pre-answer embedding is constant across prompts until attention runs.

Partial computation shows up everywhere, `control` included. The held-out named prompts carry about as much linearly-decodable result as the ceiling of the fit set (≈ 0.6 at the deep layers). The `open` conditions lift the mid-depth transfer a little, which fits the idea that the arithmetic is doing real work on these prompts, and their *final*-layer R^2 drops on the named sets, where the job of the last layer has become committing to a surface form. The main point, though, is that "computed but not emitted" was already true in `control`: making the computation stronger (`open`) or the readout available (`rev`) still does not join the two up.

The garden path, revisited

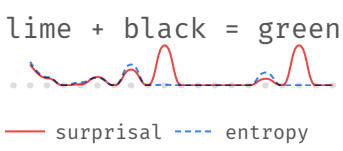
Let's return to the walkthrough example from ex-2.1.1. A garden-path sentence is one that leads you into a wrong reading until a later word forces a correction, and that's what happens here. On `lime + black = green`, the model guesses the second operand as *blue*, revises to *black* when it sees the `a`, and then only half-corrects the answer, so a memorized neighbor of the mix comes out ahead.

Which neighbor varies. At seed 0 the completions are `control` gray, `rev` gray, `open` teal, `both` teal, and across the other seeds `control` also gives olive and teal. The competitor is not one fixed wrong answer.

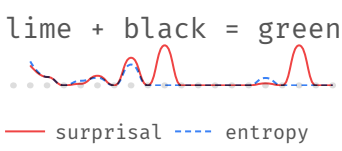
With the new corpora, does the correction finally overtake the lookup? It does not: the margin stays several nats negative. The sublines below trace per-character surprisal and entropy for seed 0.

condition	seed 0	seed 1	seed 2	mean
`control`	-11.4	-10.6	-18.4	**-13.4**
`rev`	-7.1	-9.4	-9.7	**-8.7**
`open`	-7.5	-13.9	-7.1	**-9.5**
`both`	-19.6	-11.9	-11.2	**-14.2**

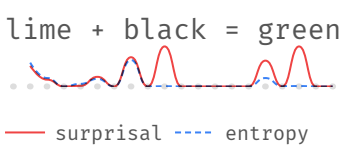
Margin of `green` over its best competitor on the `lime + black` prompt, per seed and condition. A positive value means the arithmetic comes out ahead.



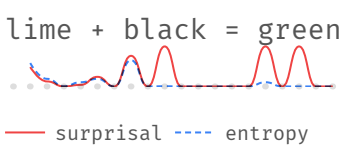
control



rev



open



both

Every set the model fails, it fails confidently rather than hedging, which is what the garden-path sublines illustrate. Across the 30 condition and eval-set pairs at zero accuracy, s_2 averages 0.61 and spans 0.41 to 0.85.

Findings

Adding the inverse dictionary and the pressure to compute on named prompts did teach the model those two skills: reverse aliases reach 1.0, and off-palette generalization reaches ≈ 0.92. But the model didn't learn to combine them: `named_holdout` stays at zero everywhere.

The model often gets the value right but writes it as hex instead of as a name, and the value → name translation never runs partway through an equation, even though the model has learned that mapping perfectly well on its own. Nothing in training ever asks the model to chain the two skills inside a single forward pass, and the just-in-time answer schedule suggests it never holds a full intermediate result anywhere as a latent embedding.

So for D2.1, `named_holdout` is not able to identify degradation: it sits at zero, so it has no headroom to lose. `open_holdout` from the new corpus may be a better indicator: it is compositional (unseen pairs, with the form of the answer decided by a computed value), and it sits near the ceiling without quite reaching it.

---

1. "Form choice margin": the log-probability of the correct hex minus that of the true name under teacher forcing. ↩