The Gradient Does Not See Rank: Rank-Indifference in Matrix-CODI on ProsQA

Samuel Larson [1]

Abstract

Continuous chain-of-thought models compress reasoning into latent tokens. Matrix-valued variants introduce rank as a single-sample structural observable on the latent matrix Z. If matrix latents carry parallel reasoning paths via superposition, rank should track them, and truncating Z to low rank should hurt accuracy on tasks whose solutions plausibly require multiple components. Across four training regimes of a matrix-CODI model (three on ProsQA, one on GSM8K-Aug below the learning threshold), the rank-k projection ablation curve is flat to within 0.6 percentage points. A three-seed replication yields 81.5 ± 1.2pp accuracy while the final effective rank of Z spans $\{4, 12, 13\}$; the loss does not reward any particular rank. To test whether rank-blindness arises from the flatten-then-project readout alone, we trained four readouts: a bilinear reparametrization, a bilinear-plus-GELU readout nonlinear in Z, an SVD-augmented readout feeding singular values through an MLP, and a quadratic readout in $ZZ^\top$. All four rank-k curves remain flat (Spearman p-values $0.63, 0.14, 0.82, 0.46$). The flat curves persist for readouts nonlinear in Z. A linear probe on Z underperforms a raw pretrained hidden state at target prediction (AUC 0.673 vs. 0.846). A negative control on vanilla GPT-2 SFT (no matrix bottleneck, no Z, three seeds, $n = 500$) reproduces a flat rank-k curve under the same intervention paradigm with pooled-mean range 0.20pp, and a random-h sensitivity floor lands at the same accuracy: the rank-k ablation alone conflates rank-blindness with position-irrelevance.

[1]Pebble Machine Learning. Correspondence to: Samuel Larson <samlarson@pebbleml.com>.

Mechanistic Interpretability Workshop at the 43rd International Conference on Machine Learning, Seoul, South Korea, 2026.
Copyright 2026 by the author(s).

1. Introduction

Continuous chain-of-thought (CoT) models replace explicit textual reasoning steps with continuous latent tokens fed back into the transformer's residual stream. COCONUT (Hao et al., 2024) and CODI (Shen et al., 2025) are representative instances: both compress an explicit rationale into a small number of continuous latent positions and decode the answer from the resulting state. Theoretical work (Zhu et al., 2025; Gozeten et al., 2025) argues that these latents can hold multiple reasoning paths in *superposition*, so continuous CoT could explore a search tree in parallel.

Rizvi-Martel et al. (2026) challenged this: a fine-tuned COCONUT model reaches 96.6% on ProsQA *without* feeding back any latent tokens, against 99.0% with latents and 85.3% for explicit CoT. They named this the *Illusion of Superposition*.

Matrix-valued latents (e.g., Shen et al., 2025) make the measurement concrete: a $d \times d$ thought Z has a computable rank via its SVD. If each singular direction encodes a separate reasoning path, truncating Z to rank k at inference should degrade accuracy when the task needs more than k paths, making the rank-k ablation curve a natural probe.

But rank counts parallel paths only under conditions the matrix parametrization does not guarantee: the stored features must align with the singular directions of $Z = \mathrm{reshape}(W_{\text{up}} h)$, and the training objective must reward that alignment. Superposition, moreover, need not be orthogonal: features packed at small mutual angles need not raise the rank, so the number of superposed features need not equal the rank. We therefore test directly whether matrix-CODI training makes rank a functional readout of reasoning. It does not.

1.1. Contributions

We report five results on a matrix-CODI bottleneck (GPT-2 small, d=16, six latent positions, ProsQA):

- Rank-k ablation is flat at two distillation weights, with the multiplicative thinker on or off, and on GSM8K-Aug as well as ProsQA. Range across $k \in \{1, 2, 4, 8, 16\}$ is ≤ 0.6pp.

- Three seeds at otherwise identical hyperparameters land at effective ranks $\{4, 12, 13\}$ and accuracy of 81.51 ± 1.2pp.
- Four nonlinear-in-Z readouts (bilinear, bilinear+GELU, SVD-augmented, quadratic in $ZZ^\top$) also give flat curves; Spearman p in $[0.14,\ 0.82]$.
- A linear probe on Z (1536 features across six positions) reaches AUC 0.673 on ProsQA target prediction; a pretrained GPT-2 hidden state at 768 features reaches 0.846 on the same task.
- Rank-k on vanilla GPT-2 SFT (no Z, three seeds) reproduces a flat curve under the same intervention paradigm; pooled range 0.20pp.

The seed-level rank spread is what distinguishes the rank-blindness reading from position-irrelevance.

2. Background

2.1. The matrix-CODI bottleneck

CODI distillation. CODI (Shen et al., 2025) trains a student to compress an explicit chain-of-thought into a fixed number of continuous latent positions by matching a hidden-state target from a teacher pass. The teacher pass consumes prompt + CoT + answer and produces a reference hidden state at a designated colon-token position. The student pass consumes prompt + n latent positions + answer, where each latent position is produced by feeding the previous step's hidden state back as the next input embedding. A hidden-state L1 loss (the *distillation* loss) aligns the student's state at the answer colon to the teacher's, and a standard next-token cross-entropy loss trains the answer prediction. The total loss is $\mathcal{L} = \gamma \mathcal{L}_{\text{kd}} + \mathcal{L}_{\text{ce}}$.

Why matrix latents. A vector of dimension D has no single-sample notion of how many independent components it superposes; a $d \times d$ matrix has rank as a structural observable. Matrix-valued memory has a long lineage *inside* attention and SSM layers (§6). Matrix-CODI is different in placement: the matrix sits on the explicit chain-of-thought feedback path, one $d \times d$ thought per latent reasoning position. We test whether the rank of that matrix behaves as a per-step count of reasoning paths under CODI-style training.

Matrix bottleneck. We extend CODI with a *matrix bottleneck* on the latent feedback path. Given the previous latent hidden state $h \in \mathbb{R}^D$ (here $D = 768$ for GPT-2 small):

$$\begin{aligned}
\text{flat} &= W_{\text{up}}\, h, \qquad W_{\text{up}} \in \mathbb{R}^{d^2 \times D} \\
Z &= \text{reshape}(\text{flat};\, d, d) \in \mathbb{R}^{d \times d} \\
Z &\leftarrow (I + \Delta(Z))\, Z\, (I + \Gamma(Z)) \quad \text{(optional thinker)} \\
h_{\text{out}} &= \text{LayerNorm}(\phi(Z)),
\end{aligned}$$

where $\phi : \mathbb{R}^{d \times d} \to \mathbb{R}^D$ is the *readout*. The default readout is $\phi(Z) = W_{\text{down}}\, \text{vec}(Z)$, which we call the *flatten-then-project* readout. $W_{\text{down}} \in \mathbb{R}^{D \times d^2}$. We use $d = 16$ throughout.

2.2. Probing rank

Rank-k ablation. At inference, compute the SVD $Z = U \Sigma V^\top$ and replace Z by its rank-k truncation

$$Z_k \;=\; U_{:,\,:k}\, \Sigma_{:k,\,:k}\, V_{:,\,:k}^\top$$

before the readout. If rank is functional, accuracy should drop as k decreases to 1. If rank is vestigial, the curve is flat.

Effective rank. The numerical rank of Z is always d since it is a dense trained matrix. For training curves we report *effective rank*, a smooth spectral proxy:

$$\begin{aligned}
H_{\text{eff}}(Z) &= \exp\left(-\sum_i \tilde{\sigma}_i \log \tilde{\sigma}_i \right), \\
\tilde{\sigma}_i &= \sigma_i \big/ \textstyle\sum_j \sigma_j .
\end{aligned}$$

The ablation itself uses hard top-k truncation. The question of interest is whether the structural capacity offered by rank > 1 is functionally used.

ProsQA. ProsQA (Hao et al., 2024) is a synthetic entailment task: a diamond-shaped directed acyclic graph of entities and a question *which leaf entity has property P?*. Each problem has a unique positive answer and a single distractor. We use the training split from the original COCONUT release. Rank-k ablations use the 128-problem test split from the original CODI release; the linear probe and negative control draw on a larger held-out set of 500 problems, noted where used.

3. The Flatten-Then-Project Readout is Rank-Blind

3.1. Four flat rank-k curves

We ran the matrix bottleneck under four training conditions that vary the task, the CODI distillation weight γ, and the

Table 1. Rank-k projection ablation across four training conditions. Z rank is the mean effective rank of the 16×16 matrix thought at eval time. r_s is the Spearman rank correlation between per-sample effective rank and correctness. Range across $k \in \{1, 2, 4, 8, 16\}$ is ≤ 0.6pp in every row. R1 (GSM8K-Aug) is at a 6% operating point below the learning threshold and is not interpretable on its own.

Run	Task	γ	Thinker	Z rank	k=1	k=16	r_s
R1	GSM8K-Aug	1.0	on	5.5	6.00%	6.12%	-0.023
R2	ProsQA	1.0	on	10.2	78.4%	78.4%	$+0.026$
R3a	ProsQA	0.0	on	12.7	76.8%	76.6%	-0.105
R3b	ProsQA	0.0	off	12.8	72.6%	72.4%	$+0.095$

multiplicative thinker. Each run produced a single rank-k ablation curve.

Varying the task (arithmetic vs. logical), the distillation weight (removing the L1-at-colon loss raises effective rank from ~ 10 to ~ 13 but leaves the curve shape unchanged), or the multiplicative thinker (off drops accuracy by ~ 1pp; curve still flat) does not bend any of the four curves. A vanilla GPT-2 SFT model with no Z at all also produces a flat curve under the same probe (§5.4). The three-seed decoupling result (Fig. 1) below is the model-level evidence the negative control cannot reproduce by construction.

3.2. Why: the readout Jacobian is constant in Z

The flatten-then-project readout is $\phi(Z) = W_{\text{down}} \operatorname{vec}(Z)$. Its Jacobian with respect to $\operatorname{vec}(Z)$ is the constant matrix W_{down}:

$$\frac{\partial \phi}{\partial \operatorname{vec}(Z)} = W_{\text{down}}.$$

By the chain rule, the loss gradient is

$$\frac{\partial \mathcal{L}}{\partial \operatorname{vec}(Z)} = W_{\text{down}}^{\top} \frac{\partial \mathcal{L}}{\partial \phi}.$$

The pullback factor $W_{\text{down}}^{\top}$ is independent of Z. Any Z-dependence of the gradient enters only through the upstream factor $\partial \mathcal{L}/\partial \phi$, which depends on Z only through the value $\phi(Z)$, not through any explicit factor of Z's SVD basis.

Proposition 1 (Linear readout has constant Jacobian)**.** *Let $\phi : \mathbb{R}^{d \times d} \to \mathbb{R}^{D}$ be linear in Z. The Jacobian $\partial \phi / \partial \operatorname{vec}(Z)$ is a constant matrix $W \in \mathbb{R}^{D \times d^2}$, and the loss gradient is $\partial \mathcal{L}/\partial \operatorname{vec}(Z) = W^{\top} \partial \mathcal{L}/\partial \phi$. The pullback factor $W^{\top}$ is independent of Z. Any rank-dependence of $\partial \mathcal{L}/\partial Z$ enters only through the upstream factor $\partial \mathcal{L}/\partial \phi$, which sees Z only through $\phi(Z)$. The chain rule therefore introduces no term that couples directly to the SVD basis of Z; the loss has no built-in rank reward. Implicit bias from the optimizer (Adam + weight decay) and upstream regularization may still shape rank through channels outside $\mathcal{L}$ (§6).*

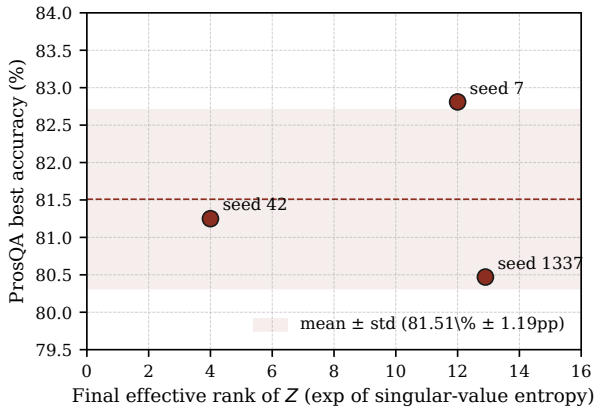

Figure 1. Three-seed replication of the flatten readout. Best ProsQA accuracy is tight at 81.51 ± 1.2pp, but the final effective rank of Z varies $3\times$: seed 42 converges at rank ~ 4; seeds 1337 and 7 converge near rank 12. The loss does not push Z toward any particular rank.

Proof sketch. Differentiating $\phi(Z) = W \operatorname{vec}(Z)$ gives $\partial \phi / \partial \operatorname{vec}(Z) = W$, a constant matrix. The chain rule then yields $\partial \mathcal{L}/\partial \operatorname{vec}(Z) = W^{\top} \partial \mathcal{L}/\partial \phi$; the prefactor $W^{\top}$ does not depend on Z. □

Remark. The proposition is a statement about the chain-rule pullback, not about the full Z-dependence of the gradient. Different Z's produce different $\partial \mathcal{L}/\partial \phi$ and therefore different gradients. What is ruled out is an SVD-basis-dependent term in the chain rule itself. Whether the absence of an explicit rank preference yields rank-blind behavior in practice is an empirical question; §5 tests it by replacing ϕ with nonlinear functions, and the flat curves persist, suggesting at least part of the explanation lies elsewhere.

The four training conditions in Table 1 are consistent with the proposition's prediction for linear ϕ. §5 tests what happens when the linearity premise is violated.

3.3. Accuracy is decoupled from rank across seeds

Proposition 1 predicts that the loss landscape should be flat along rank-changing directions of Z: a model ending training at rank 12 and a model ending at rank 4 can achieve the same accuracy. We test this with three training seeds of the same flatten-readout configuration (gpt2-small, ProsQA, $\gamma = 0$, 25 epochs, batch 16).

3.4. Linear probe on Z

If rank is vestigial, what *does* Z carry? A 5-fold cross-validated multi-class logistic regression predicts the ProsQA target class from a flattened Z on 500 held-out test problems. Controls: the same prompt's hidden state from a pretrained

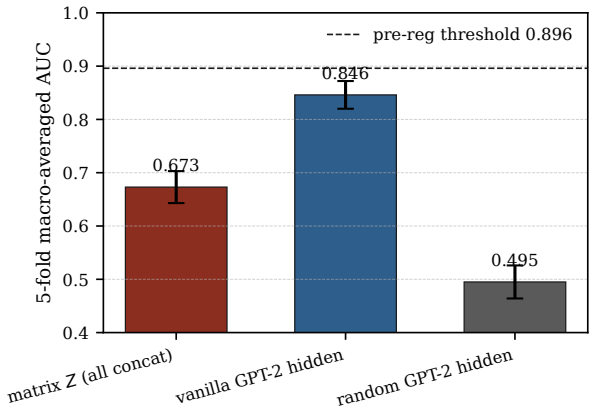

Figure 2. Linear probe AUC for ProsQA target class prediction. Pre-registered threshold for a positive result was $\max(\text{vanilla}, \text{random}) + 0.05 = 0.896$. The matrix Z concat AUC of 0.673 does not exceed it. Vanilla GPT-2, never trained on ProsQA, predicts the target class better than the trained matrix-CODI bottleneck.

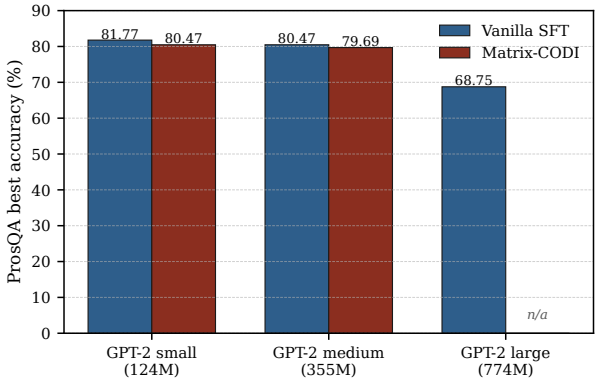

Figure 3. Scale sweep on ProsQA. Vanilla SFT degrades at GPT-2 large (68.75%) compared to GPT-2 small (81.77%, three-seed mean). ProsQA (17,886 training examples) likely under-optimizes the larger backbone at default AdamW lr $= 10^{-4}$. Matrix-CODI's best accuracy is below its matched vanilla SFT baseline at both tested scales; gaps are within three-seed standard deviation. GPT-2 large matrix-CODI is pending.

GPT-2 with no ProsQA fine-tuning, and the same model at random initialization.

Vanilla pretrained GPT-2 reaches AUC 0.846 at 768 features. The matrix-CODI bottleneck's concatenated matrix thought (6 latent positions $\times$ 256 features $=$ 1536 features) reaches AUC 0.673: more features than the vanilla hidden state, lower predictive signal for the ProsQA target. A dimension-matched comparison (probe on the post-bottleneck reconstructed 768-dim hidden state $W_{\text{down}}\,\text{vec}(Z)$ that the downstream transformer consumes) is pending. A binary target-vs-distractor probe on the same Z tensors is at chance (AUC 0.50–0.56) across all conditions.

4. Depth and Scale Do Not Rescue Matrix-CODI

The results in §3 and §5 may be specific to $d = 16$, GPT-2 small, and six latent positions. We test two axes: depth (number of iterative latent refinement steps) and backbone scale.

4.1. Depth sweep

Depth sweep (preliminary). At $n = 6$ latent refinement steps, vanilla CODI reaches 78.91% on ProsQA, $\sim$2.9pp below pure SFT. The $n \in \{16, 32, 64\}$ configurations exceeded memory at the default batch sizes; re-runs at smaller batches are in progress. A single non-baseline data point is not a depth sweep, so we report $n = 6$ as a single number and do not draw a trend.

4.2. Scale sweep

We trained vanilla SFT and matrix-CODI ($d = 16$, six latent positions, $\gamma = 0$) on ProsQA at three backbone sizes: GPT-2 small (124M), GPT-2 medium (355M), and GPT-2 large (774M). Matrix-CODI at GPT-2 large exceeded memory at batches of 2 and 4; it is omitted from Fig. 3.

Three observations from Fig. 3:

- **Matrix-CODI does not exceed vanilla SFT at any tested scale.** Gap is -1.3pp at GPT-2 small and -0.78pp at GPT-2 medium, both within the three-seed standard deviation of ± 1.2pp measured at GPT-2 small (§3). The sign is consistent across two scales. Matrix-CODI at GPT-2 large is pending.
- **Vanilla SFT itself degrades with scale on ProsQA.** The ProsQA training set (17,886 examples) is small relative to GPT-2 large's capacity; default AdamW at lr $= 10^{-4}$ likely under-optimizes the larger backbone. This is a data-size artifact of ProsQA, not a finding about superposition.
- **Matrix does not rescue the regression.** If matrix-CODI's inductive bias were relevant at scale, we would expect it to preserve or improve on its gpt2-small performance as capacity grows. It does not.

This sweep establishes that matrix-CODI does not exceed a matched vanilla SFT baseline at the tested scales and training configurations. It does not rule out settings in which matrix bottlenecks help at larger scale under different training regimes.

Sample efficiency. A sample-efficiency sweep (200, 500, 2000, 5000, 17,886 training examples) puts matrix-CODI strictly below vanilla SFT at every N below 17,886, with the gap growing as N shrinks (matrix at $N = 200$: 12.6%; vanilla at $N = 200$: 26.0%). Full sample-efficiency table is in the release.

5. Positive Control: Nonlinear Readouts Also Produce Flat Curves

Proposition 1 gives a sufficient condition for the readout Jacobian to be constant in Z. We test the prediction by training four readouts that violate it and measuring the rank-k curve on each.

5.1. Four readout variants

We replace only $\phi : \mathbb{R}^{d\times d} \to \mathbb{R}^D$ in the matrix bottleneck and leave all other training hyperparameters fixed (gpt2-small, ProsQA, $\gamma = 0$, six latent positions, $d = 16$, 25 epochs, batch 16, seed 1337, AdamW at lr $= 10^{-4}$). All four are trained at $\gamma = 0$ (no CODI L1-at-colon term), so the flat curves below are about the cross-entropy loss through the matrix bottleneck, not specifically the CODI distillation loss.

Bilinear. $\phi(Z) = W\ \mathrm{probes}(Z)$ where $\mathrm{probes}(Z)_k = u_k^\top Z v_k$ for $K = d^2$ learned probe pairs $(u_k, v_k) \in \mathbb{R}^d \times \mathbb{R}^d$. Each probe is a Frobenius inner product $\langle u_k v_k^\top, Z\rangle_F$, linear in Z. A reparametrization control: it checks that the low-rank factoring of W_{down} does not change the curve.

Bilinear+GELU. $\phi(Z) = W\ \mathrm{GELU}(\mathrm{probes}(Z))$. The GELU gates each probe by a scalar that depends on Z, making ϕ nonlinear in Z. The Jacobian is not constant, but its column space in $\mathbb{R}^{d\times d}$ is fixed (the span of the $u_k v_k^\top$); only column magnitudes vary with Z. A mild violation of Proposition 1; SVD-augmented (next) is a stronger one.

SVD-augmented. $\phi(Z) = W_{\text{down}}\,\mathrm{vec}(Z) + \mathrm{MLP}(\sigma(Z))$, where $\sigma(Z) \in \mathbb{R}^d$ is the vector of singular values fed through two dense layers with GELU. We compute $\sigma(Z)$ via `torch.linalg.svdvals`, whose backward is documented as unconditionally numerically stable (in contrast to full `torch.linalg.svd`, whose backward is unstable at near-coincident singular values). This variant explicitly exposes rank to the optimizer. SVD-augmented's best accuracy (78.12%) is below the flatten baseline (80.47%), so the optimizer is not zeroing the `sigma_proj` branch and falling back to flatten alone.

Quadratic. $\phi(Z) = W_{\text{down}}\,\mathrm{vec}\,\mathrm{concat}(ZZ^\top, Z^\top Z)$. Quadratic in Z and linear in the second-moment tensor. The gradient with respect to Z depends on Z itself.

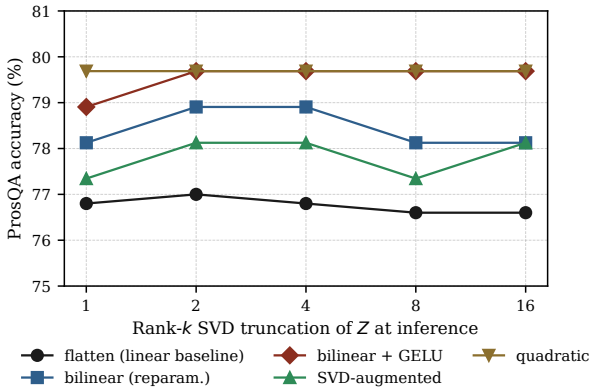

Figure 4. Rank-k projection ablation for five readouts on ProsQA. The flatten (linear) baseline is the Round 3 $\gamma = 0$ run. All four positive-control readouts, including the explicitly nonlinear-in-Z Bilinear+GELU, the SVD-augmented readout that feeds singular values through an MLP, and the quadratic readout in $ZZ^\top$, produce curves flat to within $\sim$0.8pp. The Quadratic readout is perfectly flat at 79.69% across all five k.

Table 2. Rank-k ablation accuracies (%) by readout on 128 ProsQA test problems. Spearman r_s of per-sample effective rank against correctness; p is two-sided. None significant.

Readout	k=1	k=2	k=4	k=8	k=16	r_s	p
flatten	79.0	79.0	79.0	79.0	79.0	~ 0	—
bilinear	78.1	78.9	78.9	78.1	78.1	+0.04	0.63
bilinear+GELU	78.9	79.7	79.7	79.7	79.7	−0.13	0.14
svd-aug	77.3	78.1	78.1	77.3	78.1	+0.02	0.82
quadratic	79.7	79.7	79.7	79.7	79.7	+0.07	0.46

5.2. All four rank-k curves are flat

We computed the rank-k ablation on each trained checkpoint at $k \in \{1, 2, 4, 8, 16\}$ on 128 ProsQA test problems (the standard CODI eval split).

All four p-values are above 0.14; the Quadratic readout is identical across all five k. The SVD-augmented readout, which exposes singular values directly to the optimizer, does not produce a rank-dependent curve.

5.3. The mechanism is not just readout linearity

Readouts with non-constant Jacobians still produce flat rank-k curves. Proposition 1 is sufficient but not necessary to cause them.

A plausible refinement: the trained readout's Jacobian at test inputs has an effectively rank-1 active subspace in Z regardless of whether the readout is in-principle nonlinear. In the absence of an objective term that rewards rank, every readout family tested admits a rank-1 shortcut and the optimizer takes it. We have not yet measured $\mathrm{erank}(J(Z))$ on these checkpoints to test this directly.

Table 3. Negative control: rank-k ablation on vanilla GPT-2 SFT (no matrix bottleneck, no Z). Fake Z built from $h[:256]$ reshaped to 16×16 at the six analog-latent positions. Pooled-mean range across k is 0.20pp. Per-seed Spearman r_s: $+0.32, -0.16, +0.71$ ($n=500$ test problems each).

seed	$k=1$	$k=2$	$k=4$	$k=8$	$k=16$
1337	79.80	80.20	80.00	80.20	80.00
42	79.00	78.80	78.60	78.60	79.00
7	78.00	78.00	78.00	78.00	78.20
pooled	78.93	79.00	78.87	78.93	79.07

Combined with the three-seed decoupling (effective ranks $\{4, 12, 13\}$ at matched accuracy, Fig. 1), the rank of Z in matrix-CODI is a free direction in the loss landscape, and the four readouts above do not constrain it. Candidates that might (we do not test them): an explicit rank reward in the loss, tasks with verifiable multi-rank ground truth, or step-level supervision in the spirit of SIM-CoT (Shen et al., 2026).

5.4. Negative control: rank-k ablation on vanilla GPT-2 SFT

If the flat rank-k curves were specific to the matrix-bottleneck objective, running the same probe on a model with no bottleneck should bend them. We run that test on a vanilla GPT-2 small fine-tuned for ProsQA via standard supervised fine-tuning (PURE-SFT; no latent tokens, no Z, no distillation). Three seeds $\{1337, 42, 7\}$ trained to $\sim$79pp ProsQA accuracy matching the paper's vanilla baseline. We then *construct* a fake Z at inference by reshaping the first 256 dimensions of h into a 16×16 matrix at the six token positions immediately preceding the answer-prefix colon (the analog of matrix-CODI's six latent positions), apply rank-k truncation to the fake Z via SVD, and propagate the modified residual through the remaining transformer blocks. Decoding uses no KV cache, so the intervention re-fires at every step.

The pooled-mean range across k is 0.20pp (Table 3; Fig. 5 overlays per-seed and pooled curves on the matrix-CODI flatten reference). As a sensitivity floor, replacing h at the same six positions with i.i.d. Gaussian noise matched in mean and standard deviation produces seed accuracies $\{79.6, 79.2, 78.2\}$pp, statistically indistinguishable from the unablated $\{80.0, 78.8, 78.0\}$pp. The intervention paradigm is uninformative on this model.

A flat rank-k curve is consistent with two states: a rank-blind objective, or positions that do not carry the task's information. Vanilla SFT is in the second by construction. The rank-k ablation alone cannot distinguish them.

In matrix-CODI, the bottleneck forces information through Z at those positions during training, and the trained Z reaches effective rank 12–13 at $\gamma = 0$. The seed-decoupling result (Fig. 1) is a model-level property a position-irrelevance reading would not predict.

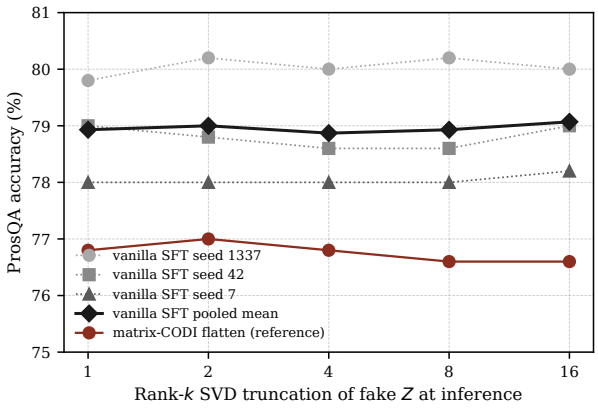

Figure 5. Negative control: rank-k ablation on vanilla GPT-2 SFT (no matrix bottleneck, no Z). Each gray dotted line is one seed of the vanilla model; the heavy black line is the pooled mean across the three seeds. The matrix-CODI flatten curve from §3 is overlaid in red as the reference. Both curves are flat to within a fraction of a percentage point.

6. Related Work

Latent CoT and the Illusion of Superposition. Rizvi-Martel et al. (2026)'s fine-tuned COCONUT reaches 96.6% on ProsQA *without* latent feedback, against 99.0% with it. Our vanilla SFT at GPT-2 small reaches 81.77%, roughly 15pp below that; we did not close the gap. The qualitative phenomenon replicates at our operating point: matrix-CODI at 82.03% and pure SFT at 81.77% differ by less than the three-seed standard deviation. The scale sweep in §4 places matrix-CODI a comparable margin below this baseline instead; in either direction the difference is within seed noise, which is the substantive point: latent feedback does not move accuracy beyond it. Our contribution relative to that work is a structural argument about the training objective: the matrix-bottleneck produces rank-indifferent gradients, and four nonlinear-in-Z readouts fail to bend the rank-k curve.

SIM-CoT. Shen et al. (2026) diagnose latent CoT instability as insufficient step-level supervision and propose injecting per-step targets. Our diagnosis is at a different layer (the matrix bottleneck's objective produces rank-indifferent gradients, adjudicated by four positive-control readouts). The two diagnoses are consistent with both mechanisms operating simultaneously.

Reasoning by Superposition and CoT2. Zhu et al. (2025) prove that a two-layer transformer with D steps

of continuous thought can solve directed graph reachability, with each thought encoding a parallel BFS frontier. Gozeten et al. (2025) show similar parallel-exploration behavior under a GRPO-style training regime. Both are theoretical capacity results with small empirical demonstrations. Capacity and what-gets-learned are distinct; our result is about what CODI distillation shapes, not whether transformers can in principle encode superposition.

February 2026 rank measurements (direct adjacency). Nazari & Rusch (2026) measure the effective rank of linear attention hidden states and propose post-training rank pruning of K and Q matrices. Sun et al. (2026) report "state-rank stratification" during pretraining: linear-attention heads bifurcate into persistently low-rank and high-rank groups. Both papers measure rank in the *fast-weight memory* inside an attention layer (a $d \times d$ accumulator), and both make *descriptive* claims about what trained networks end up with. The object of study here is different (the explicit per-position matrix latents Z on the matrix-CODI feedback path, not a fast-weight memory inside attention), and the claim is a mechanism claim about the training objective (Proposition 1) that we test by constructing four nonlinear-in-Z positive controls.

Dynamics within latent CoT. Li et al. (2026) run multiple intervention protocols on latent CoT hidden states (zero, mean, step-wise mean, Gaussian noise) and an early-stop decoding that truncates latent computation after step k. Their early-stop decoding is analogous to our rank-k ablation on a different axis: step depth vs. spectral truncation. Their step-wise causal structure is consistent with our linear-probe decay $Z[1] \to Z[5]$ (early positions carry information).

Rank-trajectory probing in depth-recurrent latent CoT. Lu et al. (2025) probe rank trajectories across recurrent blocks in Huginn-3.5B, a depth-recurrent transformer that reuses layers at inference. They find limited evidence of interpretable latent CoT via rank-trajectory analysis. Our setting differs in both model class (decoder-only GPT-2 with explicit per-position Z, not depth-recurrent shared-weight blocks) and observable (rank-k truncation of the trained Z, not the rank trajectory across recurrent depth), but the two papers reach a kindred negative reading: rank-based probing of latent CoT does not, in either setting, straightforwardly reveal a multi-path superposition picture.

Rank decay and token uniformity in stacked attention. Dong et al. (2021) showed that pure attention loses rank doubly exponentially with depth. Yan et al. (2022) characterize the related token-uniformity phenomenon in BERT-family encoders via the singular-value distribution of layer outputs and propose a transformation that flattens the spectrum. Both lines measure rank in the activations of a stack of attention layers; ours measures the rank of an explicit matrix latent on a feedback path. The two objects of study are distinct.

Implicit low-rank bias. Gradient descent on matrix-factorization losses has an implicit bias toward low-rank solutions (Gunasekar et al., 2017; Arora et al., 2019; Razin & Cohen, 2020); Kobayashi et al. (2024) attribute this specifically to weight decay. These concern bias through the parameter space, not the readout. Our three-seed rank spread $\{4, 12, 13\}$ is consistent with a weak attractor and inconsistent with strong collapse, which would concentrate all seeds at the same low rank.

Alternative substrates and probe critique. Deng et al. (2025) argue that latent reasoning lives in the vocabulary column space, not in hidden-state SVD directions; if so, rank-k truncation targets the wrong observable. Li & Janson (2024) note that zero/resample ablations (rank-k truncation is one) overestimate component importance vs. optimal ablation, which would only flatten our curves further. The negative control in §5.4 is the empirical counterpart.

7. Discussion and Limitations

Single task, single architecture family. All core experiments run on ProsQA (Hao et al., 2024) on GPT-2 {small, medium, large}. The GSM8K-Aug result in Table 1 is at a 6% operating point where the model is barely learning the task; it is not strong evidence on its own. The structural argument (Proposition 1) is stated in terms of the readout ϕ and is therefore architecture-agnostic, but the empirical evidence covers only this scale family. Cross-dataset replication on GSM8K at a higher-accuracy operating point is pending.

Seed-dependent Z rank. The three-seed decoupling (Fig. 1) is a separate finding: the same configuration, varying only the seed, produces models at effective ranks $\{4, 12, 13\}$ with accuracies $\{81.25, 82.81, 80.47\}$. Three seeds do not give statistical power to claim the rank distribution is flat or uniform; the narrower claim is that seeds at otherwise identical hyperparameters converge to materially different effective ranks, inconsistent with a strong loss-side preference for a specific rank. Implicit regularization from the optimizer (Adam + weight decay; §6) may still shape rank through channels outside $\mathcal{L}$. An $n = 10$ replication is pending.

One seed per positive control. The four positive-control variants in §5 were each trained once (compute-bounded). Spearman p-values are computed on 128 test problems per checkpoint, which limits power for small effects. A three-seed replication per variant (~ 42 H100-hours) and

re-running the four positive-control rank-k evaluations on the full 500-problem ProsQA test set are both pending; the 500-problem eval raises power to detect $|r_s| \geq 0.15$ from $\sim 40\%$ to $\sim 80\%$ at $\alpha = 0.05$.

Alternative explanation: the task is rank-1-solvable. ProsQA has a unique positive answer and a single distractor. If all answer-predictive information lives in one singular direction of Z, every architecture would converge to a rank-1 functional solution and rank-k truncation would be flat. Our data are consistent with that. What the strong reading does not explain is that the trained Z reaches effective rank 12–13 at $\gamma = 0$ instead of collapsing to 1, and three seeds spread to $\{4, 12, 13\}$ rather than concentrating. The model builds rank it does not functionally use, and the rank it builds is seed-dependent. A reasoning task whose ground truth provably requires $k > 1$ independent quantities at the answer position would disambiguate; we do not have one at this scale.

A result that would revise our reading: a readout that bends the rank-k curve on ProsQA (or a comparable structured task) under a matrix-bottleneck objective. The four readouts in §5 were chosen to maximize the chance of seeing one. A different training objective that explicitly rewards rank is a separate direction we do not address.

8. Conclusion

The matrix-bottleneck training objective in CODI does not reward rank: the readout Jacobian carries no rank information through the chain rule, the flat rank-k curves are insensitive to nonlinear readouts that escape the linear-Jacobian shortcut, and three seeds under matched hyperparameters land at effective ranks $\{4, 12, 13\}$ with statistically indistinguishable accuracy. The rank-k probe alone could not distinguish rank-blindness from position-irrelevance; the seed-level rank spread does.

9. Reproducibility

All training, evaluation, and probe code is released at https://github.com/saml212/matrix-codi-rank-blindness. The release includes:

- `run_matrix_codi.py`: the matrix-CODI training and rank-k evaluation script. The `MatrixBottleneck` class implements the $W_{\text{up}} \rightarrow$ reshape $\rightarrow$ thinker $\rightarrow$ flatten $\rightarrow W_{\text{down}}$ pipeline; all five readouts in §5 are selectable via the `--readout` flag. The same script computes the accuracy-vs-k curve and per-sample Spearman correlation between effective rank and correctness on a saved checkpoint.
- `probe_z.py`: the linear probe pipeline for §3. Produces the AUC numbers in Fig. 2.
- Raw rank-k evaluation output files (JSON) for the four positive-control readouts, matching Table 2 and Fig. 4.
- A human-readable experiment log with per-run hyperparameters, rank trajectories, and wall-clock times.

Datasets: ProsQA (Hao et al., 2024) and GSM8K-Aug (Shen et al., 2025), both from the original releases.

Backbone: pretrained GPT-2 small (124M), medium (355M), and large (774M) from the standard public checkpoints.

Training hardware: a single NVIDIA H100 (80GB HBM3) per run. All reported numerical results in the main body trace to a specific checkpoint and evaluator run in the release.

Headline numbers by source. Table 1 rows R1–R3b come from `run_matrix_codi.py` at seed 1337; Fig. 1 from the same script at seeds 1337, 42, and 7. Table 2 and Fig. 4 are from `run_matrix_codi.py` run once per readout (flatten, bilinear, bilinear+GELU, SVD-augmented, quadratic), evaluated via the same script's rank-k mode. Fig. 2 is from `probe_z.py` on the Round 3 $\gamma = 0$ checkpoint. The preliminary depth result in §4 and Fig. 3 are from the vanilla SFT and vanilla CODI scripts in the same release.

Impact Statement

This paper presents a negative empirical result on a specific mechanistic-interpretability probe (the rank-k ablation curve applied to matrix-valued continuous chain-of-thought latents under CODI-style distillation). The work advances the field of machine learning by clarifying that one widely cited single-sample structural observable does not, in this training regime, measure what its name suggests. There are many potential societal consequences of advancing machine-learning interpretability, none of which we feel must be specifically highlighted here.

References

Arora, S., Cohen, N., Hu, W., and Luo, Y. Implicit regularization in deep matrix factorization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. arXiv:1905.13655.

Deng, J., Pang, L., Wei, Z., Xu, S., Duan, Z., Xu, K., Song, Y., Shen, H., and Cheng, X. LLM latent reasoning as

chain of superposition. *arXiv preprint arXiv:2510.15522*, 2025.

Dong, Y., Cordonnier, J.-B., and Loukas, A. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. In *International Conference on Machine Learning (ICML)*, 2021. arXiv:2103.03404.

Gozeten, A., Ildiz, M. E., Zhang, Y., Harutyunyan, H., Rawat, A. S., and Oymak, S. Continuous chain of thought enables parallel exploration and reasoning. *arXiv preprint arXiv:2505.23648*, 2025.

Gunasekar, S., Woodworth, B., Bhojanapalli, S., Neyshabur, B., and Srebro, N. Implicit regularization in matrix factorization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. arXiv:1705.09280.

Hao, S., Sukhbaatar, S., Su, D., Li, X., Hu, Z., Weston, J., and Tian, Y. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

Kobayashi, S., Akram, Y., and von Oswald, J. Weight decay induces low-rank attention layers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. arXiv:2410.23819.

Li, M. and Janson, L. Optimal ablation for interpretability. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. arXiv:2409.09951.

Li, Z., Bai, X., Chen, K., Li, Y., Yang, J., Lin, C., and Zhang, M. Dynamics within latent chain-of-thought: An empirical study of causal structure. *arXiv preprint arXiv:2602.08783*, 2026.

Lu, W., Yang, Y., Lee, K., Li, Y., and Liu, E. Latent chain-of-thought? decoding the depth-recurrent transformer. *arXiv preprint arXiv:2507.02199*, 2025.

Nazari, P. and Rusch, T. K. The key to state reduction in linear attention: A rank-based perspective. *arXiv preprint arXiv:2602.04852*, 2026.

Razin, N. and Cohen, N. Implicit regularization in deep learning may not be explainable by norms. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. arXiv:2005.06398.

Rizvi-Martel, M., Rabusseau, G., and Mosbach, M. The illusion of superposition? a principled analysis of latent thinking in language models. *arXiv preprint arXiv:2604.06374*, 2026.

Shen, Z., Yan, H., Zhang, L., Hu, Z., Du, Y., and He, Y. CODI: Compressing chain-of-thought into continuous space via self-distillation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2025. arXiv:2502.21074.

Shen, Z. et al. SIM-CoT: Step-level implicit supervision for continuous chain of thought. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2509.20317.

Sun, A., Zhang, H., Zhou, H., Ma, Y., Qin, Y., Su, T., Liu, Y., Ma, Z., Xu, J., Gao, J., Hao, J., and He, R. State rank dynamics in linear attention LLMs. *arXiv preprint arXiv:2602.02195*, 2026.

Yan, H., Gui, L., Li, W., and He, Y. Addressing token uniformity in transformers via singular value transformation. *arXiv preprint arXiv:2208.11790*, 2022.

Zhu, H., Hao, S., Hu, Z., Jiao, J., Russell, S., and Tian, Y. Reasoning by superposition: A theoretical perspective on chain of continuous thought. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2505.12514.