STRUCTURING NEURAL REPRESENTATIONS WITH GEOMETRIC SUPERVISION

Anonymous authors
Paper under double-blind review

ABSTRACT

Can mechanistic understanding of language models enable training with targeted supervision of internal representations? We leverage previously uncovered neural geometry to design training objectives that sharpen the representations of task-relevant variables. First, we fine-tune Llama-3.1-8B to answer questions such as "What month is four months after June?" by directly supervising representations of the input month (June = 6) and offset (4), which are encoded with Fourier harmonics in the residual stream. Second, we fine-tune the model to predict line breaks in text wrapped to a fixed width by directly supervising the number of characters since the preceding newline, which is represented by a low-dimensional helical structure. In both experiments, we achieve parity with standard fine-tuning using only internal supervision with no output-level task loss. Moreover, the success of geometric supervision can not be isolated to the targeted representations because patching in those representations is not sufficient to recover the performance gain. Together, these findings show how mechanistic understanding of neural geometry can inform training.

1 INTRODUCTION

A central promise of interpretability is that mechanistic understanding can lead to more surgical training objectives that directly target concepts latent within neural networks (Geiger et al., 2022; Wu et al., 2022; Meng et al., 2022; Zur et al., 2024; Wu et al., 2024; Prakash et al., 2025; Cho et al., 2025; Bai et al., 2025; Chen et al., 2025; Bergen et al., 2026; Casademunt et al., 2025; Yin et al., 2024). In this work, we design training objectives that target low dimensional geometric structure in internal representations that has an established algorithmic role (e.g., Engels et al. 2025; Kantamneni & Tegmark 2025; Zhou et al. 2024; Gurnee et al. 2025; Feucht et al. 2026; Sarfati et al. 2026; Wurgaft et al. 2026).

We study Llama-3.1-8B on questions such as "What month is four months after June?" and on predicting line breaks in text wrapped to a fixed width. These tasks build on mechanistic accounts of base-10 addition (Feucht et al., 2026) and character counting (Gurnee et al., 2025). For questions about months, we supervise the input month (June $= 6$) and offset (4), which are represented through base-10 Fourier features used in an addition computation. For linebreaking, we supervise the number of characters since the preceding newline, drawing on the helical structure of character-count representations. In each case, we train low-rank adapters (LoRAs; Hu et al. 2022) with a loss that aligns representations with an idealized geometric target. Across the two tasks, we find that internal geometric supervision achieves similar performance to standard fine-tuning despite no supervision on outputs.

We further investigate why geometric supervision is successful and uncover a surprising finding: while both standard fine-tuning and geometric supervision sharpen the existing task-relevant geometry, when we patch in either sharpened representations back into the original model, we do not recover the performance gain. This demonstrates that geometric supervision produces beneficial effects that cannot be isolated to the supervised representation achieving the idealized geometry.

Our results simultaneously provide a functional validation of the mechanistic accounts of these behaviors provided by (Feucht et al., 2026) and (Gurnee et al., 2025) and prove out a pathway for knowledge of neural geometry and computation to enable surgical control via targeted training updates.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

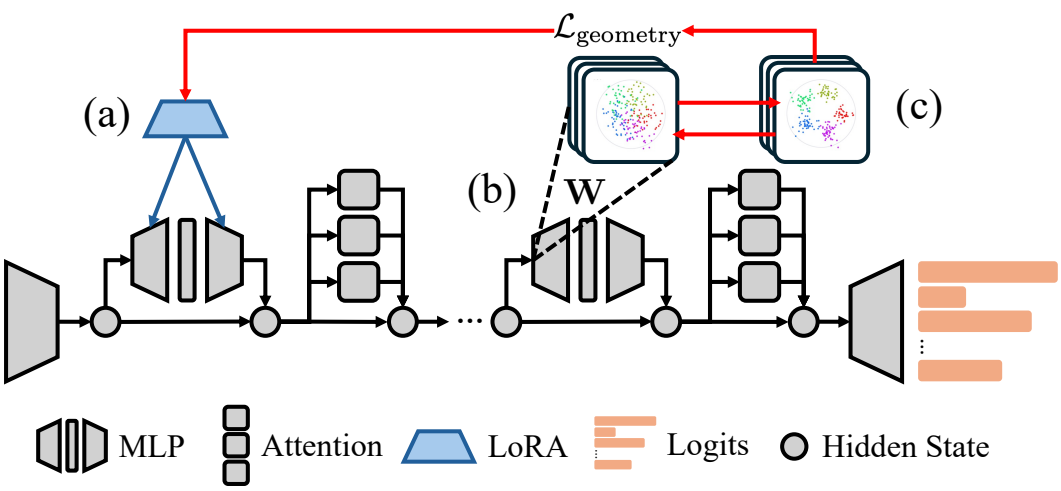


Figure 1: An overview of geometric supervision. **(a)** A LoRA is placed at an early layer with the goal of aligning empirical geometric structure to an idealized target geometry. **(b)** Supervision on the task-relevant geometry is applied at an intermediate layer in the k-dimensional coordinates of $\mathbf{W}$. **(c)** The geometry LoRA is trained by minimizing a loss (Equation 2) that aligns the observed structure $\hat{\mathbf{G}}$ to the target structure $\mathbf{G}$, without any output-level task loss.

2 Methodology

Notation. Let f_θ be a transformer language model (LM) with parameters θ, where θ_0 denotes frozen base model parameters. Let X be a set of prompts corresponding to a given task. Each prompt $x \in X$ is associated with a set of task variables $\mathcal{C}$, where each variable $c \in \mathcal{C}$ is a function $c : X \to \mathcal{V}_c$ that maps a prompt x to a value $v \in \mathcal{V}_c$. For a given variable c, we denote the set of prompts that share the value $v \in \mathcal{V}_c$ as $X_v = \{x \in X : c(x) = v\}$. Let $\mathbf{h}_x(\theta) \in \mathbb{R}^d$ represent a hidden state activation extracted from the model's computation $f_\theta(x)$ under prompt x. For each task we investigate, prior work has identified task-relevant activation geometry that can be read out of the hidden state by a map $\phi : \mathbb{R}^d \to \mathbb{R}^k$, which we call a *readout*. In both tasks studied here, ϕ is affine, $\phi(\mathbf{h}) = \mathbf{A}\mathbf{h} + \mathbf{b}$, and we measure geometry in the readout coordinates. The activation that a given prompt x takes on under the readout can be written as $\phi(\mathbf{h}_x(\theta))$, and for prompts that share a value v, we write the average readout as:

$$\hat{\boldsymbol{\mu}}_v(\theta) = \frac{1}{|X_v|} \sum_{x \in X_v} \phi(\mathbf{h}_x(\theta)). \tag{1}$$

Our Approach. We choose tasks that have previously identified task-relevant geometric structure with the goal of using that structure to fix model behavior. Rather than train on the model's output, our approach consists of fine-tuning a model with supervision that aligns empirical geometric structure with an idealized target geometry (see Figure 1). Each target structure is task-dependent (e.g., points equally spaced on a circle or helix) and will be described in more detail for each task in Sections 3.2 and 4.2.

Specifically, given a model f_θ with observed geometric structure $\hat{\mathbf{G}} = \{\hat{\boldsymbol{\mu}}_v\}_{v \in \mathcal{V}_c} \subset \mathbb{R}^k$, that has been observed under ϕ, we supervise the model's representations to match an "idealized" target geometric structure $\mathbf{G} = \{\boldsymbol{\mu}_v\}_{v \in \mathcal{V}_c} \subset \mathbb{R}^k$. The training objective we use for geometric alignment between $\hat{\mathbf{G}}$ and $\mathbf{G}$ consists of two loss terms:

$$\mathcal{L}_{\text{geometry}} = \mathcal{L}_{\text{mean}} + \lambda \mathcal{L}_{\text{spread}} \tag{2}$$

where λ is a hyperparameter balancing the two terms. The first term, $\mathcal{L}_{\text{mean}}$, pulls each empirical centroid $\hat{\boldsymbol{\mu}}_v$ toward its target position $\boldsymbol{\mu}_v$, as seen in Equation 3:

$$\mathcal{L}_{\text{mean}} = \frac{1}{|\mathcal{V}_c|} \sum_{v \in \mathcal{V}_c} \frac{1}{r_0^2} \left\| \hat{\boldsymbol{\mu}}_v(\theta) - \boldsymbol{\mu}_v \right\|_2^2 \tag{3}$$

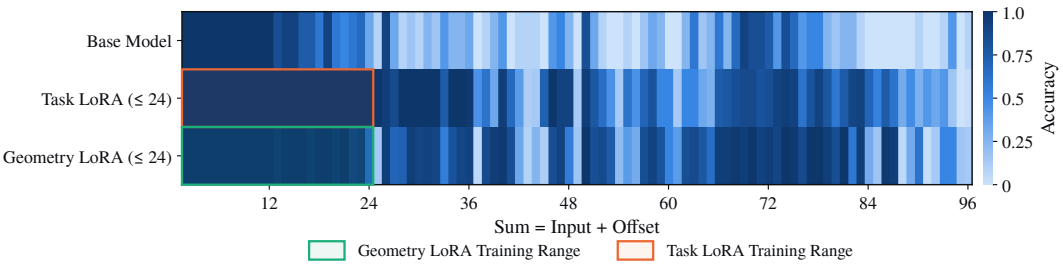


Figure 2: Performance of Llama-3.1-8B and fine-tuned variants on the months task where inputs are grouped by sum, e.g., "What is four months after June?" is grouped with "What is nine months after January?" for having the same sum of 10. The base model (top row) achieves 100% accuracy when the $\texttt{sum} \leq 12$, but performance degrades for larger sums, achieving a total accuracy of only 36.3%. Training a LoRA with end-to-end supervision ("Task LoRA", middle row) on sums up to 24 improves performance to 71.9%, with the strongest gains primarily located within the training range ($X_{\texttt{sum}\leq 24}$, gray shaded region) and the first held-out cycle ($\texttt{sum} \in [25, 36]$). Training a LoRA with *only* geometric supervision (§3.2) over the same training range ("Geometry LoRA", bottom row) achieves similar improvements across the board, with a total accuracy of 73.3%.

where $r_0^2 = \frac{1}{|\mathcal{V}_c|} \sum_{v \in \mathcal{V}_c} ||\hat{\boldsymbol{\mu}}_v(\theta_0) - \bar{\boldsymbol{\mu}}_c(\theta_0)||_2^2$ normalizes the loss by the variance of the frozen base model's centroids around their mean $\bar{\boldsymbol{\mu}}_c(\theta_0) = \frac{1}{|\mathcal{V}_c|} \sum_{v \in \mathcal{V}_c} \hat{\boldsymbol{\mu}}_v(\theta_0)$.

The second loss term, $\mathcal{L}_{\text{spread}}$, targets a ratio τ between the fine-tuned and base-model's within-value spread (see Appendix A.4 for an ablation study), shown in Equation 4:

$$\mathcal{L}_{\text{spread}} = \left(\frac{\sigma_c(\theta)}{\sigma_c(\theta_0)} - \tau \right)^2 \tag{4}$$

where $\sigma_c(\theta)$ is the root-mean-square (RMS) distance of projected activations to their value centroids:

$$\sigma_c(\theta) = \left(\frac{1}{|X|} \sum_{v \in \mathcal{V}_c} \sum_{x \in X_v} \|\phi(\mathbf{h}_x(\theta)) - \hat{\boldsymbol{\mu}}_v(\theta)\|_2^2 \right)^{1/2}. \tag{5}$$

A ratio of $\tau = 1$ would encourage the fine-tuned model's projected activation spread to match the spread of the base model exactly, whereas $\tau > 1$ would increase the spread in the fine-tuned model.

3 Supervising Fourier Features for Month Arithmetic

The first task we study is the **months task** from Feucht et al. (2026). An individual prompt for the months task takes the form $x =$"*Q: What month is {offset} months after {input}?\nA:*", and the goal of the model is to predict the correct month. The months task has three variables: $\mathcal{C} = \{\texttt{input}, \texttt{offset}, \texttt{sum}\}$, and the set of prompts X for this task is constructed by cycling through the values of two variables—the input and the offset, which determine the third sum variable (i.e., $\texttt{sum} = \texttt{input} + \texttt{offset}$). The input takes on values in $\mathcal{V}_{\text{input}} = \{\text{January, February, ... December}\}$ (where January is 1-indexed), and the offset takes on values in $\mathcal{V}_{\text{offset}} = \{1, 2, ..., 95\}$ such that X includes all prompts where $\texttt{sum} \leq 96$ (i.e., $\mathcal{V}_{\text{sum}} = \{2, 3, ..., 96\}$), giving a total of $|X| = 1074$ prompts. We note that Feucht et al. only evaluate the model on prompts where $\texttt{offset} \leq 24 = 2p$ (so that $\texttt{sum} \leq 36$), where $p = 12$ is the cycle length of the input concept, but we include a larger range of offset values to test for broader generalization (i.e., $\texttt{sum} \leq 96 = 8p$).

Evaluation of end-to-end fine-tuning. We evaluate Llama-3.1-8B (Llama Team, 2024) on the months dataset ($X_{\texttt{sum}\leq 96}$) and find that it achieves 100% accuracy when the value of sum is small (≤ 12), but quickly degrades as the value of sum increases, achieving an overall accuracy of 36.3% (Figure 2, gray dashed line). We test whether fine-tuning the model can correct these errors by training a low-rank adapter (LoRA; Hu et al., 2022) using a standard cross-entropy loss between the model's prediction and the correct month. We use $X_{\texttt{sum}\leq 24}$ as the training set, and evaluate on all prompts in $X_{\texttt{sum}\leq 96}$. With the LoRA, the model achieves a total accuracy of 71.9% (Figure 2,

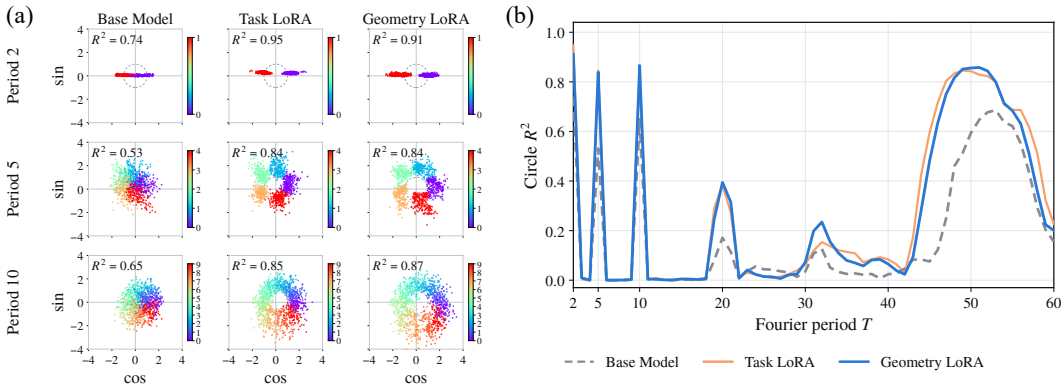


Figure 3: **(a)** Projection of the months data from the base model, task LoRA, and geometry LoRA onto planes containing Fourier features for the sum variable with periods $T = \{2, 5, 10\}$ at layer 18. Each point corresponds to a prompt $x \in X_{\texttt{sum}\leq 96}$, and is colored by $\texttt{sum}(x) \bmod T$. The LoRA sharpens existing base-10 Fourier representations, resulting in tighter per-modulo clusters, and higher alignment with idealized Fourier structure. The geometry LoRA (§3.2) directly supervises the representations to produce similar sharpening. **(b)** Alignment of representations with ideal Fourier planes for periods $T \in \{2, ..., 60\}$. Base-10 periods sharpen under the task LoRA (orange) and geometry LoRA (blue) compared to the base model (gray), while other periods do not.

orange), fixing almost all errors within the LoRA's training distribution (Figure 2, shaded region), and also improving accuracy on the first held-out cycle ($\texttt{sum}\in[25, 36]$). Here, we report the results of the best task LoRA, but include results and training details for other configurations in Appendix A.1.

3.1 End-to-end LoRAs Strengthen Existing Representations

To understand what has changed in the model under LoRA fine-tuning, we analyze the model's task-relevant geometry. Specifically, Feucht et al. (2026) has identified a set of Fourier features with base-10 periods (e.g., 2, 5, 10, etc.) at layer 18 that Llama-3.1-8B uses to compute the sum variable for both cyclic arithmetic and standard addition tasks. We train a set of Fourier probes for each period $T \in \{2, 5, 10, 20, 50, 100\}$ for both sine and cosine at each model layer l. The probes are trained on a simple addition task ("a+b=") to match either $\sin(\frac{2\pi c(x)}{T})$ or $\cos(\frac{2\pi c(x)}{T})$, where $c \in \{a, b, a+b\}$ is the variable being probed for (see Appendix A.2 for details). For a prompt x, we can project the final token hidden state $\mathbf{h}_x(\theta)$ onto the Fourier plane for variable c and period T at layer l, denoted as $\phi_c^{(l,T)}$, by applying the corresponding trained probes:

$$\phi_c^{(l,T)}(\mathbf{h}_x(\theta)) = (\mathbf{w}_{\text{cos}}^{(l,T)} \cdot \mathbf{h}_x(\theta) + \mathbf{b}_{\text{cos}}^{(l,T)}, \mathbf{w}_{\text{sin}}^{(l,T)} \cdot \mathbf{h}_x(\theta) + \mathbf{b}_{\text{sin}}^{(l,T)}) \in \mathbb{R}^2 \tag{6}$$

where $\mathbf{w}_{\text{cos}}^{(l,T)}, \mathbf{w}_{\text{sin}}^{(l,T)} \in \mathbb{R}^d$ and $\mathbf{b}_{\text{cos}}^{(l,T)}, \mathbf{b}_{\text{sin}}^{(l,T)} \in \mathbb{R}$ are the probe weights.

We inspect the geometry by projecting the months data onto each Fourier plane and find that the circular structure of the base-10 periods is visibly sharper in the LoRA fine-tuned models compared to the base model (Figure 3a; see also Appendix A.2). We measure the R^2 value of the sum Fourier planes and find that for the LoRA fine-tuned models, the R^2 value of the base-10 periods ($T \in \{2, 5, 10, 20, 50\}$) increases on average by 0.23 at layer 18, while the average change of other periods is $\sim$0.04 (Figure 3b). An increased alignment of base-10 periods with the ideal Fourier geometry and a minimal effect on other periods suggests that LoRA fine-tuning might achieve at least part of its performance gain by strengthening the existing representational structure.

3.2 Geometric Supervision on Internal Representations

Motivated by the sharpening of existing representational structure seen in Section 3.1, we investigate whether targeting geometric structure directly can provide a sufficient learning signal to correct the model's behavior without a loss on the model output. Training a LoRA with only a geometric loss requires choosing: (1) where to supervise the geometry (e.g., an affine projection of the residual stream), and (2) what the geometry should look like (i.e., an idealized target geometry $\mathbf{G}$).

We choose to apply geometric supervision at the final-token residual stream at layer $l^* = 18$, the layer at which Feucht et al. (2026) localize the `sum` computation based on the `input` and `offset` representations.[1] Rather than a single projection, we use a set of Fourier planes, one for each pair (c, T) with $c \in \mathcal{C}' = \{\texttt{input}, \texttt{offset}\}$ and $T \in \mathcal{T} = \{2, 5, 10, 20\}$. This gives eight planes with total dimension $k = 16$ captured by the orthogonal columns of the matrix $\mathbf{W}$:

$$\mathbf{W}\mathbf{h}_x(\theta) \;=\; \Big[\, \phi_c^{(l^*,T)}(\mathbf{h}_x(\theta)) \,\Big]_{(c,T)\,\in\,\mathcal{C}'\times\mathcal{T}} \;\in\; \mathbb{R}^k,\; k = 2|\mathcal{C}'||\mathcal{T}|. \tag{7}$$

We train a LoRA with geometric supervision by minimizing the loss in Equation 2. Each loss term is computed for each plane separately and then averaged over all eight planes.

Computing the empirical structure and target structure for Fourier features. Consider a plane $\phi_c^{(l^*,T)}$ containing the Fourier features for variable c and period T. The empirical centroids $\hat{\boldsymbol{\mu}}_v(\theta)$ are computed using Equation 1, for each value v that c can take over the training distribution (i.e., our training data realizes only a subset of values $\mathcal{V}_c' \subseteq \mathcal{V}_c$). The empirical structure is the set of these centroids, $\hat{\mathbf{G}}_{(c,T)}(\theta) = \{\hat{\boldsymbol{\mu}}_v(\theta)\}_{v\in\mathcal{V}_c'} \subset \mathbb{R}^2$. For the same (c, T) Fourier plane, the idealized target structure places each value v on a ring: the target position of v on the period-T plane is a point $\mathbf{u}_v$ on the unit circle at phase $2\pi v/T$:

$$\mathbf{u}_v = \left(\cos\left(\tfrac{2\pi v}{T}\right),\; \sin\left(\tfrac{2\pi v}{T}\right)\right) \in \mathbb{R}^2. \tag{8}$$

This ensures that values with the same residue $v \bmod T$ share the same target, e.g., $3 \bmod 5$ and $8 \bmod 5$. A Fourier plane is only defined up to rotation and reflection, so we align the ideal pattern $\{\mathbf{u}_v\}_{v\in\mathcal{V}_c'}$ to the base model's per-value centroids $\{\hat{\boldsymbol{\mu}}_v(\theta_0)\}_{v\in\mathcal{V}_c'}$ via orthogonal Procrustes alignment (Schönemann, 1966) where the target centroid $\boldsymbol{\mu}_v$ and structure $\mathbf{G}_{(c,T)}$ are computed as:

$$\boldsymbol{\mu}_v = \bar{\boldsymbol{\mu}}_c(\theta_0) + \alpha_r\, r_0\, \mathbf{Q}_0\, \mathbf{u}_v, \qquad G_{(c,T)} = \{\boldsymbol{\mu}_v\}_{v\in\mathcal{V}_c'} \subset \mathbb{R}^2 \tag{9}$$

where $\mathbf{Q}_0 \in O(2)$ is a rotation-reflection matrix, and α_r scales the target radius relative to the base model's centroid spread r_0. We fit $\mathbf{Q}_0$ once before training, and keep the target centroids $\mathbf{G}_{(c,T)}$ fixed throughout.

Geometric supervision achieves parity with end-to-end supervision. As with the task LoRA, we use $X_{\texttt{sum}\leq 24}$ as the training set for our geometry LoRA (see Appendix A.3 for other configurations), and evaluate on all prompts in $X_{\texttt{sum}\leq 96}$. As seen in Figure 2, the geometry LoRA (shown in blue) improves over the base model across the board, achieving a total accuracy of 73.3%. It scores 37.0% higher than the base model and is on-par with the best task LoRA (71.9%) trained on the same data, despite never training with loss on model outputs.

3.3 Sharpened Fourier features are not sufficient

The geometry LoRA's training objective directly supervises the model's representations to produce a similar sharpening effect to the one observed under the task LoRA (Figure 3, right). However, even though both task and geometry fine-tuning successfully *sharpen* these representations, we find that simply patching these activations back into the base model to "clean up" its geometry is not sufficient to recover the gains made by either fine-tuning objective.

Specifically, we test this by replacing the base model's activations with those of a fine-tuned model only within the supervised Fourier planes (i.e. the coordinates under $\mathbf{W}$) for each prompt. For an individual Fourier plane, we write this intervention as: $\phi_c^{(l,T)}(\mathbf{h}_x(\theta_0)) \leftarrow \phi_c^{(l,T)}(\mathbf{h}_x(\theta))$. When we perform this intervention on the supervised Fourier planes, it has almost no effect on the base model's accuracy (from 36.3% to 37.3%); patching a random subspace of the same rank has the same null effect. This result holds for patching from either an end-to-end LoRA or a geometry LoRA. By comparison, patching the 16 directions along which the fine-tuned and base model activations differ the most at recovers nearly all of the improvement. This demonstrates that there are beneficial off-target effects of fine-tuning that are not captured by representations targeted for supervision. Fixing the geometry of planes from Fourier probes is not sufficient to fix the model's behavior, but a loss that trains the model to fix the geometry is successful.

[1] For more training details, including other supervision configurations, see Appendix A.3.

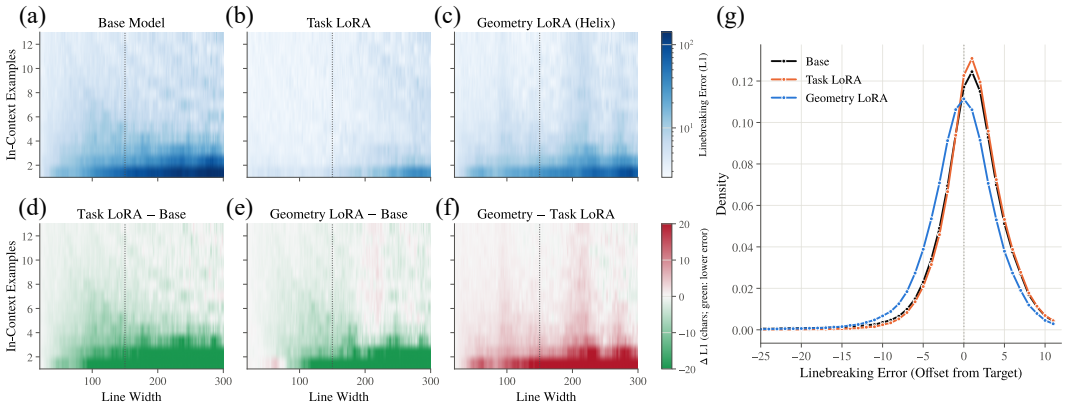


Figure 4: Evaluating on the linebreaking task. **Top:** Linebreaking error (L1) in characters as a function of line width (x-axis) and number of in-context examples (y-axis) for **(a)** the base Llama-3.1-8B model (left; mean L1 of 10.7 characters), **(b)** the end-to-end LoRA (center; 4.5) and **(c)** geometry LoRA (right; 7.4). **Bottom:** Difference in linebreaking error between two models (ΔL1 capped at [-20,20]), where green indicates the first model has lower error, while red indicates the second model is better. **(d)** The task LoRA and **(e)** the geometry LoRA improve over the base model at nearly every cell (97% and 88%, respectively), with the strongest improvements in short contexts of long widths. **(f)** Compared to each other, the task LoRA has lower linebreaking error almost everywhere, with larger gaps in shorter contexts ($1-3$ in-context examples), while the difference in linebreaking error is near-parity (within $\sim$0.77 characters) with five or more in-context examples. **(g)** Distribution of linebreaking error in characters. Most linebreaking errors are concentrated near the optimal target width.

4 Supervising Helical Structure for In-Context Linebreaking

The second task we study is the **linebreaking task** from Gurnee et al. (2025) where LMs must learn to produce lines of text that wrap according to a maximum width of characters. We construct prompts for this task by taking documents from FineWeb (Penedo et al., 2024), stripping their newlines, and reinserting a newline (\n) at the nearest word boundary that will produce a line of at most m characters for $m \in \{1, ..., 300\}$.[2] For a wrapped document, we treat each wrap point (including its preceding lines of text) as a separate prompt x. Given a passage of text where all lines are wrapped to a fixed width m, the model must predict a newline at the appropriate width (see Appendix B.1 for examples). A wrap point appearing later in a document is preceded by many earlier lines of the same width, acting as in-context examples of when to linebreak. This task has a single variable: $\mathcal{C} = \{\texttt{character-count}\}$, where $\texttt{character-count}(x_t) \leq m$ is the number of characters since the previous newline at the token position t of prompt x.

For evaluation, we construct a dataset of 300 documents that have been wrapped to widths of $m \in \{1, ..., 300\}$, which we denote as $X_{m \leq 300}$. To correctly predict a newline at width m, the model must keep track of the character-count and predict either words that will lead to a character count less than m or a newline character \n. We measure performance in terms of *linebreaking error*—which is the number of characters by which a predicted linebreak misses the target width m, accounting for the length of the next word. A predicted linebreak at token t is considered valid if the sum of the $\texttt{character-count}(x_t)$ and the length of the next predicted word $(\text{len}(x_{t+1}))$ is greater than or equal to the target width m (i.e., $\texttt{character-count}(x_t) + \text{len}(x_{t+1}) \geq m$).

Evaluation of task fine-tuning. We evaluate Llama-3.1-8B on our linebreaking dataset $X_{m \leq 300}$, and find that the base model attains a linebreaking error of 10.7 characters on average (Figure 4a). Error is bigger with fewer in-context examples and at larger widths. As before, we test whether fine-tuning can correct the base model's errors by training a task LoRA with a cross-entropy loss on the newline prediction using a training set with widths $m \leq 150$ (i.e., $X_{m \leq 150}$; see Appendix B.2 for more details). The task LoRA reduces the base model's linebreaking error to an average of 4.5

[2]For values of $m \leq 5$, this produces text where a newline is inserted after almost every word.

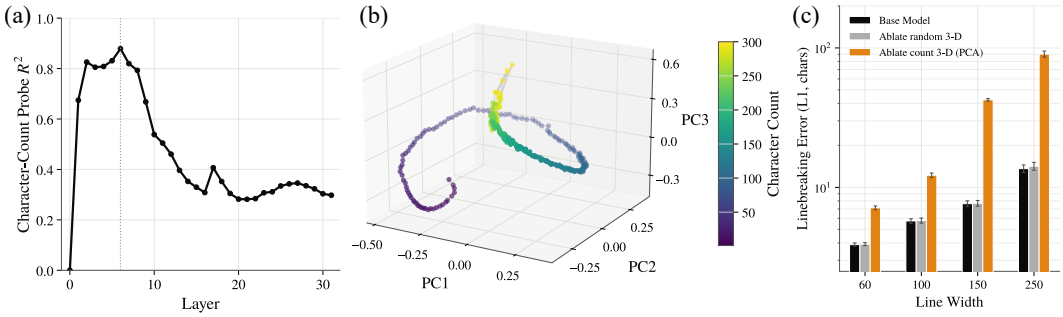


Figure 5: Characterizing the internal representation of `character-count`. **(a)** Fitting a linear regression probe at each layer of Llama-3.1-8B to predict `character-count`. The probe achieves its highest score ($R^2 = 0.88$) in early layers ($l \approx 6$). **(b)** Average `character-count` activations projected via PCA follow a helix-like pattern. **(c)** Ablating the top-3 PCA directions that contain the `character-count` manifold at layer 6 sharply increases linebreaking error at each width tested (e.g., from 7.6 to 42.5 characters at width m=150). Ablating a random 3-dimensional subspace has no effect on the error.

characters (Figure 4b), and improves error in 97% of cells (Figure 4d). The largest gains are in short contexts (1−3 in-context examples), where the linebreaking error drops from 32.9 to 6.8 characters.

For each prompt, we take the model's *first* predicted newline and record its offset from the target width: $\texttt{character-count}(x_t) + \text{len}(x_{t+1}) - m$, pooled over all depths and widths and plot the distributions in Figure 4g. Values below 0 are errors, while values ≥ 0 are valid linebreaks (the next word would overflow m). We find that most linebreak errors for each model are still concentrated near the optimal target width. The task LoRA (orange) makes linebreak predictions too early 29.9% of the time, and the base model (black) is similar (33.9% too early).

4.1 Characterizing the Internal Representation of Character Count

Seeing that Llama-3.1-8B can perform the linebreaking task well, we ask how it represents the `character-count` variable, and whether it matches the helix-like manifold reported by Gurnee et al. (2025) in their study of the closed-weight model Claude 3.5 Haiku. Following Gurnee et al. (2025), we characterize the structure by: (1) training linear-regression probes $\mathbf{w}^l_{\texttt{count}}$ on each layer's residual stream to predict $\texttt{character-count}(x_t)$, (2) applying PCA to the per-`character-count` mean activations to visualize the geometry, and (3) running interventions to test whether the geometry is causal.

We fit linear-regression probes at each layer of Llama-3.1-8B to predict the `character-count` variable. Probes are fit on tokens from FineWeb documents wrapped to widths m=150 over counts 20–150. We find that the `character-count` is linearly decodable, as linear probes achieve high accuracy ($R^2 = 0.88$) at early layers ($l \approx 6$) while scores drop off in later layers (see Figure 5a).

We compute the average layer-6 residual stream activation for each value of $\texttt{character-count} \in [1, 150]$ and apply PCA and find the top 3 PCs explain 92.5% of the means' variance. To visualize their structure, we project the data onto the top 3 PCs. We find that the means lie on a helix-like structure in the first 3 PCs (Figure 5b): two components rotate with the `character-count` variable, while a third axis increases almost monotonically as `character-count` increases. To validate that this geometry is causally relevant for linebreaking, we ablate the top-3 PCA components containing the `character-count` manifold at layer 6 and test its effect on newline prediction. For comparison, we also ablate the activations within a random 3-dimensional subspace. We find that ablating a random 3-dimensional subspace has little to no effect on the model's performance while ablating the count subspace increases linebreaking error by a large margin—for example, increasing from 7.6 to 42.5 characters at width m=150 (Figure 5c; see also Appendix B.4).

Effect of task fine-tuning on the count manifold. As in the months task (§3.1), we ask whether the task LoRA fixes newline errors by strengthening the geometry the model already has. We fit an idealized helix to the layer-6 hidden states of all tokens, aligning each token to the ideal position for

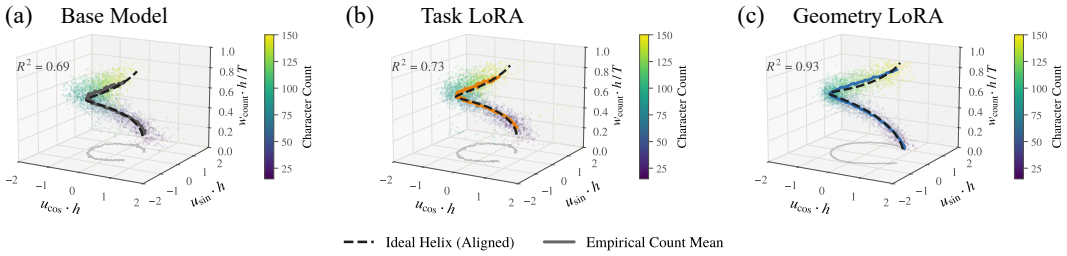


Figure 6: The layer-6 `character-count` manifold under the **(a)** base model, **(b)** task LoRA, and **(c)** geometry LoRA. We measure the fraction of variance an idealized helix captures over all tokens (R^2). Plotted points are individual token hidden states projected onto a fixed helix (§ 4.2) and colored by `character-count` $\in [20, 150]$. The solid curve in each plot is the empirical per-`character-count` mean and the dashed curve is the idealized helix. The base model is already well-aligned with an ideal helix (R^2=0.69), and task fine-tuning sharpens it slightly (R^2=0.73). Direct geometric supervision aligns the manifold the most (R^2=0.93).

its `character-count` by orthogonal Procrustes—and report the fraction of variance it captures (R^2; Appendix B.4). We find that the base model already follows the helix well (R^2=0.69; Figure 6a), and task fine-tuning has a small positive improvement (R^2=0.73; Figure 6b).

4.2 Geometric Alignment on Helical Structure

Similar to Section 3.2, we train a geometry LoRA that targets the helical structure directly, without any output-level task loss. We supervise the final-token residual stream at layer l^*=6, where the `character-count` variable is most strongly represented (Figure 5a). In this case, the projection $\mathbf{W} \in \mathbb{R}^{3\times d}$ is represented as a k=3 dimensional *helix*: two dimensions are Fourier probes $\mathbf{w}_{\cos}, \mathbf{w}_{\sin}$ trained with the `character-count` variable and period T, and the third dimension is the `character-count` probe $\mathbf{w}_{\texttt{count}}$:

$$\mathbf{W}\mathbf{h}_x(\theta) = \left(\mathbf{w}_{\cos} \cdot \mathbf{h}_x(\theta) + \mathbf{b}_{\cos},\ \mathbf{w}_{\sin} \cdot \mathbf{h}_x(\theta) + \mathbf{b}_{\sin},\ \mathbf{w}_{\texttt{count}} \cdot \mathbf{h}_x(\theta) + \mathbf{b}_{\texttt{count}}\right) \in \mathbb{R}^3. \quad (10)$$

The empirical structure $\hat{\mathbf{G}}(\theta) = \{\hat{\boldsymbol{\mu}}_v(\theta)\}_{v\in\mathcal{V}'_{\texttt{count}}}$ is the set of per-`character-count` centroids (Equation 1), and the idealized target structure $\mathbf{G}_{\text{helix}} = \{\boldsymbol{\mu}_v\}_{v\in\mathcal{V}'_{\texttt{count}}}$ places each `character-count` value v on the ideal helix point $\mathbf{u}_v = (\cos(2\pi v/T),\ \sin(2\pi v/T),\ v/T)$. We align the ideal helix to the base model's per-`character-count` centroids $\{\hat{\boldsymbol{\mu}}_v(\theta_0)\}$ via orthogonal Procrustes once before training (see Equation 9) and keep $\mathbf{G}_{\text{helix}}$ fixed throughout. We train the geometry LoRA by minimizing the geometric loss of Equation 2 over the same width range as the task LoRA ($m \leq 150$; see Appendix B.3 for more details).

Evaluating geometric supervision for the linebreaking task. As seen in Figure 4c, we find that the geometry LoRA is able to improve the base model's linebreaking error from 10.7 to an average of 7.4 characters, improving over the base model in 88% of (width $\times$ in-context) cells (Figure 4e). The geometry LoRA does have higher error overall compared to the task LoRA, but converges to parity with more in-context examples—a difference in linebreaking error of only 0.77 characters for contexts with five or more examples (Figure 4f). We also see that the incorrect linebreak predictions the geometry LoRA makes are concentrated near the optimal target width (Figure 4g). In this setting, the geometry LoRA reshapes the `character-count` manifold more than task supervision, aligning to the ideal helix at R^2=0.93 vs. 0.73 for the task LoRA and 0.69 for the base model (Figure 6c).

5 Related Work

Geometry of arithmetic and counting. Our work builds on accounts of base-10 arithmetic and character counting in pretrained language models (Feucht et al., 2026; Gurnee et al., 2025), and others who study arithmetic and identify Fourier features or helical representations that enable numeric computation in both small transformers and LLMs (Nanda et al., 2023; Zhong et al., 2023; Engels et al., 2025; Kantamneni & Tegmark, 2025; Zhou et al., 2024). Our work is also related to Sinii & Balagansky (2026) and Dunefsky et al. (2026) who replicate counting manifolds in open-weight models and study related countdown circuitry.

Representation geometry and control. Our work relates to geometric analyses that relate internal representations to semantic structure and beliefs about latent states (Park et al., 2024; 2025; Shai et al., 2024; Sarfati et al., 2026), though rather than a purely descriptive account, we use idealizations of those representations as a training signal. Steering methods change predictions by adding activation vectors or moving representations along learned manifolds, providing tests of how these structures affect behavior (Turner et al., 2023; Rimsky et al., 2024; Wurgaft et al., 2026).

Supervision of internal variables. Our work relates to prior methods that supervise concepts or their causal effects to improve task behavior (Koh et al., 2020; Geiger et al., 2022; Wu et al., 2022; Zur et al., 2024). In particular, recent work from Bai et al. (2025) and Zhou et al. (2026) have improved learning of arithmetic by supervising intermediate sums or designing Fourier number embeddings.

Training from internal targets. Our work is also related to Low-Rank Representation Adaptation (LoRRA) (Zou et al., 2023) which train from losses computed over hidden states, Attention Behavior Fine-Tuning Cho et al. (2025) who supervise attention scores to perfect induction heads, and (Akyürek et al., 2024) who train n-gram heads into existing LMs.

Fine-tuning with mechanistic knowledge. Our findings align with prior work showing that fine-tuning strengthens computations already present in pretrained models (Prakash et al., 2024; Jain et al., 2024). Also related are mechanistic-inspired fine-tuning methods such as ReFT and LoFiT learn changes to selected representations, and Constructive Circuit Amplification restricts weight updates to components identified through interventions (Wu et al., 2024; Yin et al., 2024; Prakash et al., 2025). Other approaches intervene on concept directions during training to shape generalization or model traits (Casademunt et al., 2025; Chen et al., 2025).

Causal evaluation. Our work relies on prior mechanistic account of task variables which are dependent on causal analysis frameworks such as Causal mediation and causal abstraction (Vig et al., 2020; Finlayson et al., 2021; Todd et al., 2024; Stolfo et al., 2023; Geiger et al., 2021; 2025; Mueller et al., 2026), which evaluate proposed mechanisms through interventions on internal states. Work on factual editing also finds that causal-tracing scores poorly predict which layers support successful weight edits (Hase et al., 2023).

6 Discussion

We have studied two cases where prior work identified causally-relevant geometric structure that an LLM uses to compute its answer: Fourier features for month arithmetic (Feucht et al., 2026), and helical representations for predicting linebreaks in structured text (Gurnee et al., 2025). While previous mechanistic accounts of these variables describe an idealized structure, the model's actual representations are noisier. We asked whether training these representations toward their ideal geometry can improve the model's behavior without directly supervising it. We found that a loss defined only on the geometry of an intermediate representation, with no output-level supervision, can achieve parity with standard fine-tuning on these tasks. This result also provides a functional validation of the mechanistic accounts we build on.

More broadly, a major goal of interpretability research is that understanding a neural network should allow us to predict when it will fail to generalize, and to surgically edit its representations or weights to fix those errors. While this is a hard problem in general, the tasks we study are easy to verify and have plentiful training data, which allows us to compare geometric supervision directly against output-level supervision. Our findings show that behavioral errors on these tasks can be corrected by specifying what the model's representations should look like, rather than what its outputs should be. This suggests that a detailed mechanistic account of a particular behavior can be used not only to describe it, but also to inform targeted training for more surgical control.

Extending geometric supervision to more settings will require methods for automatically discovering causally-relevant target geometries, choosing idealizations, successfully characterizing causally-relevant task variables mechanistically.

DISCLOSURE OF AI USE

We used generative AI tools to write and organize experiment code for all experiments, and to obtain feedback on experiments. AI was used to help write plotting code to generate figures for the paper, and to write code for building the synthetic linebreaking dataset (§ 4). AI was also used to help draft portions of the appendix, check definitions and equations, and help polish writing in the main paper. We have reviewed all AI contributions to this work and take responsibility for its final content, including its text, methods, and results.

REFERENCES

Ekin Akyürek, Bailin Wang, Yoon Kim, and Jacob Andreas. In-context language learning: Architectures and algorithms. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=3Z9CRr5srL.

Xiaoyan Bai, Itamar Pres, Yuntian Deng, Chenhao Tan, Stuart Shieber, Fernanda Viégas, Martin Wattenberg, and Andrew Lee. Why can't transformers learn multiplication? reverse-engineering reveals long-range dependency pitfalls, 2025. URL https://arxiv.org/abs/2510.00184.

Leon Bergen, Usha Bhalla, Sidharth Baskaran, Max Loeffler, Raphael Sarfati, Dhruvil Gala, Ryan Panwar, Santiago Aranguri, Thomas Fel, Atticus Geiger, Matthew Kowal, Siddharth Boppana, Daniel Balsam, Owen Lewis, Jack Merullo, Thomas McGrath, and Ekdeep Singh Lubana. Anatomy of post-training: Using interpretability to characterize data and shape the learning signal, 2026. URL https://arxiv.org/abs/2606.12360.

Helena Casademunt, Caden Juang, Adam Karvonen, Samuel Marks, Senthooran Rajamanoharan, and Neel Nanda. Steering out-of-distribution generalization with concept ablation fine-tuning, 2025. URL https://arxiv.org/abs/2507.16795.

Runjin Chen, Andy Arditi, Henry Sleight, Owain Evans, and Jack Lindsey. Persona vectors: Monitoring and controlling character traits in language models, 2025. URL https://arxiv.org/abs/2507.21509.

Hakaze Cho, Peng Luo, Mariko Kato, Rin Kaenbyou, and Naoya Inoue. Mechanistic fine-tuning for in-context learning. In Yonatan Belinkov, Aaron Mueller, Najoung Kim, Hosein Mohebbi, Hanjie Chen, Dana Arad, and Gabriele Sarti (eds.), *Proceedings of the 8th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pp. 330–357, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-346-3. doi: 10.18653/v1/2025.blackboxnlp-1.21. URL https://aclanthology.org/2025.blackboxnlp-1.21/.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Jacob Dunefsky, Wes Gurnee, and Emmanuel Ameisen. A shared subcircuit lets llms count down across tasks, 2026. URL https://arxiv.org/abs/2607.12279.

Joshua Engels, Eric J. Michaud, Isaac Liao, Wes Gurnee, and Max Tegmark. Not all language model features are one-dimensionally linear, 2025. URL https://arxiv.org/abs/2405.14860.

Sheridan Feucht, Tal Haklay, Usha Bhalla, Daniel Wurgaft, Can Rager, Raphaël Sarfati, Jack Merullo, Thomas McGrath, Owen Lewis, Ekdeep Singh Lubana, Thomas Fel, and Atticus Geiger. Arithmetic in the wild: Llama uses base-10 addition to reason about cyclic concepts, 2026. URL https://arxiv.org/abs/2605.01148.

Matthew Finlayson, Aaron Mueller, Sebastian Gehrmann, Stuart Shieber, Tal Linzen, and Yonatan Belinkov. Causal analysis of syntactic agreement mechanisms in neural language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 1828–1843. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.acl-long.144. URL https://aclanthology.org/2021.acl-long.144.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal abstractions of neural networks. In *Advances in Neural Information Processing Systems*, volume 34, pp. 9574–9586, 2021. URL https://arxiv.org/abs/2106.02997.

Atticus Geiger, Zhengxuan Wu, Hanson Lu, Josh Rozner, Elisa Kreiss, Thomas Icard, Noah Goodman, and Christopher Potts. Inducing causal structure for interpretable neural networks. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 7324–7338. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/geiger22a.html.

Atticus Geiger, Duligur Ibeling, Amir Zur, Maheep Chaudhary, Sonakshi Chauhan, Jing Huang, Aryaman Arora, Zhengxuan Wu, Noah Goodman, Christopher Potts, and Thomas Icard. Causal abstraction: A theoretical foundation for mechanistic interpretability. *Journal of Machine Learning Research*, 26(83):1–64, 2025. URL https://jmlr.org/papers/v26/23-0058.html.

Wes Gurnee, Emmanuel Ameisen, Isaac Kauvar, Julius Tarng, Adam Pearce, Chris Olah, and Joshua Batson. When models manipulate manifolds: The geometry of a counting task. *Transformer Circuits Thread*, 2025. URL https://transformer-circuits.pub/2025/linebreaks/index.html.

Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 17643–17668. Curran Associates, Inc., 2023. doi: 10.52202/075280-0774. URL https://papers.nips.cc/paper_files/paper/2023/hash/3927bbdcf0e8d1fa8aa23c26f358a281-Abstract-Conference.html.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Samyak Jain, Robert Kirk, Ekdeep Singh Lubana, Robert P. Dick, Hidenori Tanaka, Tim Rocktäschel, Edward Grefenstette, and David Krueger. Mechanistically analyzing the effects of fine-tuning on procedurally defined tasks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=A0HKeKl4Nl.

Subhash Kantamneni and Max Tegmark. Language models use trigonometry to do addition, 2025. URL https://arxiv.org/abs/2502.00873.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5338–5348. PMLR, 2020. URL https://proceedings.mlr.press/v119/koh20a.html.

AI @ Meta Llama Team. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 36, 2022. arXiv:2202.05262.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Byj72udxe.

Aaron Mueller, Jannik Brinkmann, Millicent Li, Samuel Marks, Koyena Pal, Nikhil Prakash, Can Rager, Aruna Sankaranarayanan, Arnab Sen Sharma, Jiuding Sun, Eric Todd, David Bau, and Yonatan Belinkov. The quest for the right mediator: Surveying mechanistic interpretability for NLP through the lens of causal mediation analysis. *Computational Linguistics*, 52(1):331–378, 2026. doi: 10.1162/coli.a.572. URL https://aclanthology.org/2026.cl-1.10/.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations*. OpenReview.net, 2023. URL https://openreview.net/forum?id=9XFSbDPmdW.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 39643–39666. PMLR, 2024. URL https://proceedings.mlr.press/v235/park24c.html.

Kiho Park, Yo Joong Choe, Yibo Jiang, and Victor Veitch. The geometry of categorical and hierarchical concepts in large language models. In *The Thirteenth International Conference on Learning Representations*. OpenReview.net, 2025. URL https://openreview.net/forum?id=bVTM2QKYuA.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=n6SCkn2QaG.

Nikhil Prakash, Tamar Rott Shaham, Tal Haklay, Yonatan Belinkov, and David Bau. Fine-tuning enhances existing mechanisms: A case study on entity tracking. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=8sKcAWOf2D.

Nikhil Prakash, Donghao Ren, Dominik Moritz, and Yannick Assogba. Constructive circuit amplification: Improving math reasoning in llms via targeted sub-network updates, 2025. URL https://arxiv.org/abs/2512.16914.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15504–15522. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.828. URL https://aclanthology.org/2024.acl-long.828/.

Raphaël Sarfati, Eric Bigelow, Daniel Wurgaft, Jack Merullo, Atticus Geiger, Owen Lewis, Tom McGrath, and Ekdeep Singh Lubana. The shape of beliefs: Geometry, dynamics, and interventions along representation manifolds of language models' posteriors, 2026. URL https://arxiv.org/abs/2602.02315.

Peter H. Schönemann. A generalized solution of the orthogonal Procrustes problem. *Psychometrika*, 31(1):1–10, 1966.

Adam S Shai, Sarah E Marzen, Lucas Teixeira, Alexander G Oldenziel, and Paul M Riechers. Transformers represent belief state geometry in their residual stream. In *Advances in Neural Information Processing Systems*, volume 37, pp. 75012–75034, 2024. URL https://arxiv.org/abs/2405.15943.

Viacheslav Sinii and Nikita Balagansky. Chasing the counting manifold in open llms, 2026. URL https://huggingface.co/spaces/t-tech/manifolds.

Alessandro Stolfo, Yonatan Belinkov, and Mrinmaya Sachan. A mechanistic interpretation of arithmetic reasoning in language models using causal mediation analysis. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7035–7052. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.435. URL https://aclanthology.org/2023.emnlp-main.435/.

Eric Todd, Millicent Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. Function vectors in large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=AwyxtyMwaG.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering, 2023. URL https://arxiv.org/abs/2308.10248.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. Investigating gender bias in language models using causal mediation analysis. In *Advances in Neural Information Processing Systems*, volume 33, pp. 12388–12401. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/92650b2e92217715fe312e6fa7b90d82-Paper.pdf.

Zhengxuan Wu, Atticus Geiger, Josh Rozner, Elisa Kreiss, Hanson Lu, Thomas Icard, Christopher Potts, and Noah D. Goodman. Causal distillation for language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4288–4295, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.318. URL https://aclanthology.org/2022.naacl-main.318.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D Manning, and Christopher Potts. ReFT: Representation finetuning for language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=fykjplMc0V.

Daniel Wurgaft, Can Rager, Matthew Kowal, Vasudev Shyam, Sheridan Feucht, Usha Bhalla, Tal Haklay, Eric Bigelow, Raphael Sarfati, Thomas McGrath, Owen Lewis, Jack Merullo, Noah Goodman, Thomas Fel, Atticus Geiger, and Ekdeep Singh Lubana. Manifold steering reveals the shared geometry of neural network representation and behavior, 2026. URL https://arxiv.org/abs/2605.05115.

Fangcong Yin, Xi Ye, and Greg Durrett. Lofit: Localized fine-tuning on llm representations, 2024. URL https://arxiv.org/abs/2406.01563.

Ziqian Zhong, Ziming Liu, Max Tegmark, and Jacob Andreas. The clock and the pizza: Two stories in mechanistic explanation of neural networks. In *Advances in Neural Information Processing Systems*, volume 36, pp. 27223–27250. Curran Associates, Inc., 2023. doi: 10.52202/075280-1186. URL https://papers.nips.cc/paper_files/paper/2023/hash/56cbfbf49937a0873d451343ddc8c57d-Abstract-Conference.html.

Tianyi Zhou, Deqing Fu, Vatsal Sharan, and Robin Jia. Pre-trained large language models use fourier features to compute addition, 2024. URL https://arxiv.org/abs/2406.03445.

Tianyi Zhou, Deqing Fu, Mahdi Soltanolkotabi, Robin Jia, and Vatsal Sharan. FoNE: Precise single-token number embeddings via Fourier features. In *International Conference on Learning Representations*, volume 2026, pp. 28485–28516, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/30253ebdaa75eed08c4b10c75aa8f781-Abstract-Conference.html.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to AI transparency. *arXiv preprint arXiv:2310.01405*, 2023. URL https://arxiv.org/abs/2310.01405.

Amir Zur, Elisa Kreiss, Karel D'Oosterlinck, Christopher Potts, and Atticus Geiger. Updating CLIP to prefer descriptions over captions. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 20178–20187, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.1125. URL https://aclanthology.org/2024.emnlp-main.1125.

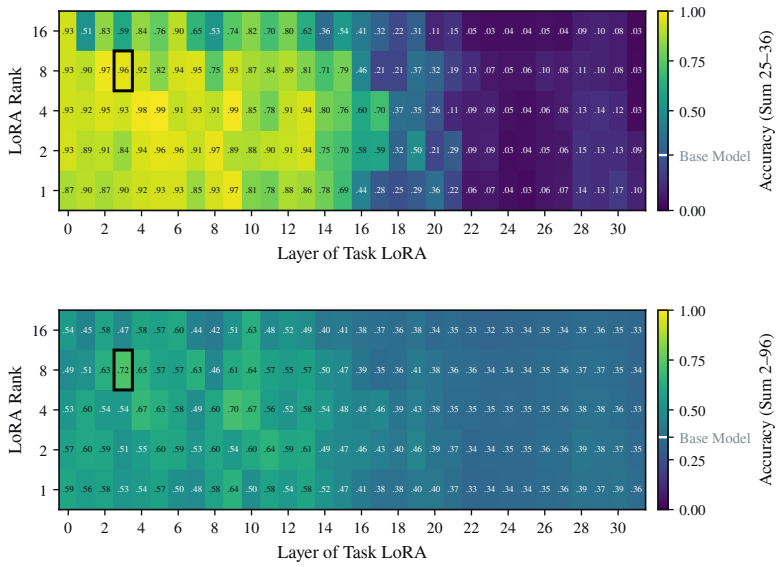


Figure 7: Comparing performance of task LoRAs trained at different layers ($0-31$) with different ranks ($\{1,2,4,8,16\}$). We measure performance on the first held-out cycle: $X_{\texttt{sum}\in[25,36]}$ (top), and the full dataset $X_{\texttt{sum}\leq 96}$ (bottom). The base model scores 27.8% on the held-out cycle range. We find that training a task LoRA at early layer MLPs ($l \leq 14$) for any rank ≤ 8 is able to fix almost all of the errors the base model makes. After about layer 14, training a task LoRA no longer meaningfully improves model performance compared to the base model's accuracy. The task LoRA configuration reported in the main paper (layer 3, rank 8) is shown with a black box.

A Months Task: Experimental Details

A.1 Training Task LoRAs

In this section, we provide additional details about how we train LoRAs with end-to-end supervision, as well as several ablations that measure how hyperparameters like location, rank, training data range, etc. affect the generalization performance of task LoRAs. All task LoRAs are trained with a standard cross-entropy loss at the final token position between the model's prediction and the correct month.

We train task LoRAs at the MLP of a single layer (i.e., the gate, up, and down projections), the LoRA update: $\Delta = s\,(xA^\top)B^\top$ with $s = 2$, is applied at every token position. We use the AdamW optimizer (Loshchilov & Hutter, 2019) with a learning rate of 2×10^{-3}, weight decay of 0.01, gradient-norm clipping at 1.0, and a constant learning rate schedule, and we train for 200 steps over the training set: $X_{\texttt{sum}\leq 24}$. The base model is Llama-3.1-8B in `bf16` and the weights are kept frozen, while the LoRA parameters are kept in `fp32`. The task LoRA results reported in the main paper use a rank-8 LoRA trained at the layer-3 MLP which reaches 71.9% accuracy on the months task.

We score a prediction as correct if the highest-scoring token among the twelve month tokens is the correct month. We use a restricted argmax to the twelve month tokens to simplify evaluation, but also find that the model's top predicted token is a month token for 99.3% of prompts.

Training layer and rank. In Figure 7, we investigate the effect of training LoRAs of different ranks ($\{1, 2, 4, 8, 16\}$) and at different locations (i.e., the MLP of layer $l \in \{0, ...31\}$), averaged over three seeds. The task LoRA configuration used in the main paper (layer 3, rank 8) is shown with a black box. For each layer/rank combination, we evaluate the LoRAs on the first held-out cycle: $X_{\texttt{sum}\in[25,36]}$, and compare to the Llama-3.1-8B's performance of 27.8% on this same range. We find that training a low-rank task LoRA at a range of early layer MLPs ($l \leq 14$) is sufficient to solve the majority of the hold-out prompts ($95-100\%$). However, task LoRAs trained at later layers do not meaningfully improve task performance beyond the base model's accuracy. We found this

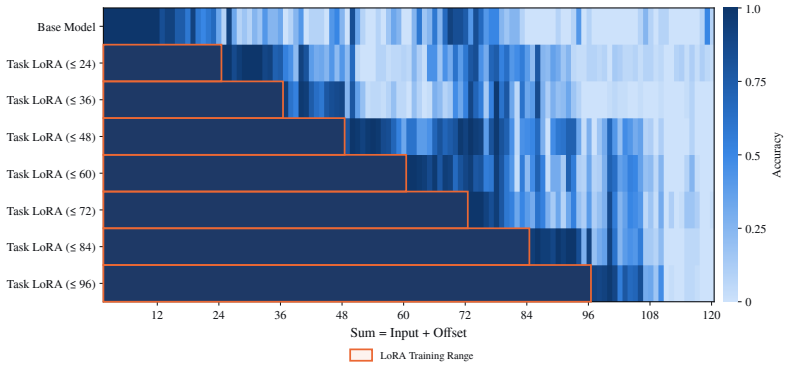


Figure 8: Performance of Llama-3.1-8B on the months task, grouped by sum. The model achieves 100% on the task when the sum ≤ 12, but performance drops for larger values. Training a LoRA with end-to-end supervision (i.e., "Task LoRA") on a subset of data in cycle-length increments ($X_{\texttt{sum}\leq k}$) can correct errors within its training range (shown in orange), but the generalization is shallow, often only up to one held-out cycle beyond the training range.

early/late distinction surprising, as prior work (Feucht et al., 2026) hypothesized that some of the model's errors were due to incorrectly computing the mod, which would be most likely to happen in later layers after the sum computation (which was previously localized to layer 18).

Table 1: Comparing single layer vs. multi-layer LoRA training. Each LoRA reported here is rank-1 and trained on $X_{\texttt{sum}\leq 24}$ with end-to-end supervision. We report the accuracy in three evaluation bands. The first band is in-range: sum $\in [2, 24]$, the other two are held-out ranges: sum $\in [25, 48]$, and sum $\in [49, 96]$. Scores are mean $\pm$ standard deviation over 3 seeds. Multi-layer LoRAs use a learning rate of 5×10^{-4}.

				Accuracy (%)		
Objective	Location	Layers	# Params.	sum $\in [2, 24]$	sum $\in [25, 48]$	sum $\in [49, 96]$
–	Base Model	–	0	79.0%	22.9%	27.4%
Task CE	MLP	4	55K	$100.0 \pm 0.0\%$	$68.2 \pm 7.8\%$	$30.7 \pm 4.2\%$
Task CE	MLP	0–16	0.94M	$100.0 \pm 0.0\%$	$70.8 \pm 3.1\%$	$31.0 \pm 4.4\%$
Task CE	MLP	0–31	1.77M	$100.0 \pm 0.0\%$	$60.9 \pm 2.3\%$	$38.5 \pm 2.9\%$
Task CE	MLP + Attn	0–31	2.62M	$100.0 \pm 0.0\%$	$55.7 \pm 3.1\%$	$31.3 \pm 5.2\%$

Task LoRA on a single layer vs. every layer. In Table 1, we measure the effect of training LoRAs on multiple layers has compared to training a LoRA on just a single layer. For the task LoRAs, we report either a single LoRA trained at the layer 4 MLP, or many LoRAs trained at layers $0-16$, $0-31$, and $0-31$ including attention layers. We find that when training LoRAs with task supervision, adding more parameters/layers does not consistently improve performance compared to training a single LoRA. The only noticeable improvement is that training on layer $0-31$ MLPs improves accuracy on the sum $\in [49, 96]$ range by $\sim 8\%$, but this also hurts performance on the sum $\in [25, 48]$ range by the same amount. This suggests that adding trainable parameters does not consistently improve generalization on this task.

Training data range. In Figure 8, we show the results of training LoRAs on increasing amounts of data in cycle-length increments (i.e., $X_{\texttt{sum}\leq 24}$, $X_{\texttt{sum}\leq 36}$, ..., $X_{\texttt{sum}\leq 96}$). We find that fine-tuning with end-to-end supervision "fixes" almost all errors *within* the LoRA's training distribution, but that the generalization to unseen data is shallow, often only extending for up to one held-out cycle beyond the training range. Because we see the same generalization pattern for each training range we test, we use a single fixed training range ($X_{\texttt{sum}\leq 24}$) for other experiments to ensure matched comparison.

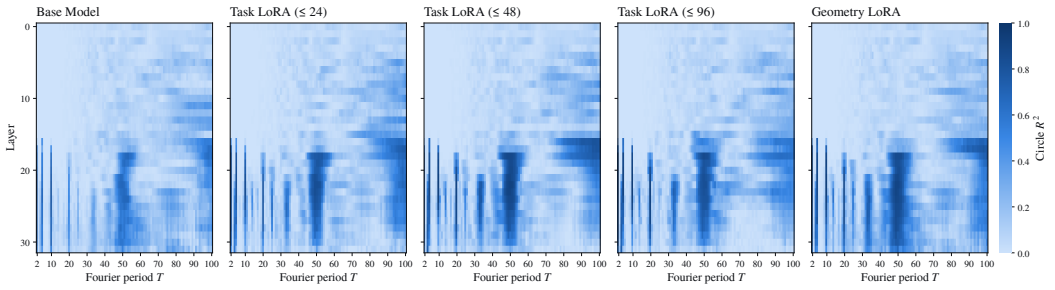


Figure 9: LoRA fine-tuning sharpens existing base-10 Fourier representations. We report the R^2 score of Fourier planes for all layers and across all periods $T \in \{2, ..., 100\}$ for Llama-3.1-8B and several LoRAs. Each subplot shows the R^2 value at a layer (row) and period T (column) for the base model or a fine-tuned variant. Roughly the same periods have high R^2 values in each in case, with fine-tuned variants having stronger peaks around the previously identified base-10 periods: $\{2, 5, 10, 20, 50\}$.

A.2 Fourier Probes

Training Fourier Probes. We follow Feucht et al. (2026) and train Fourier probes using a simple addition task where prompts have the following template $x =$"*a*+*b*=". The full probe training dataset X_{a+b} is constructed by varying a and b over each possible value of $a, b \in \{1, ..., 199\}$. For a given layer l in the model f_θ, and for each period $T \in \{2, ..., 100\}$, we train two affine probes to match the target sine and cosine harmonics. Each probe is trained by minimizing their corresponding MSE objective over X_{a+b}:

$$\text{MSE}\left(\langle \mathbf{w}_{\sin}^{(l,T)}, h_x \rangle + b_{\sin}^{(l,T)}, \sin\left(\frac{2\pi c(x)}{T}\right)\right), \ \text{MSE}\left(\langle \mathbf{w}_{\cos}^{(l,T)}, h_x \rangle + b_{\cos}^{(l,T)}, \cos\left(\frac{2\pi c(x)}{T}\right)\right). \quad (11)$$

where $\mathbf{w}_{\cos}^{(l,T)}, \mathbf{w}_{\sin}^{(l,T)} \in \mathbb{R}^d$ and $b_{\cos}^{(l,T)}, b_{\sin}^{(l,T)} \in \mathbb{R}$ and c is the variable being probed. We train distinct probes for each value of c, which in this case, we have $c \in \{a, b, a + b\}$.

When selecting probes for the months task, we use the Fourier probe trained at the corresponding variable slot. To probe for representational structure of the `input` variable, we use probes trained with $\sin\left(\frac{2\pi a}{T}\right)$ or $\cos\left(\frac{2\pi a}{T}\right)$ as the target. To probe for representational structure of the `offset` variable, we use probes trained with $\sin\left(\frac{2\pi b}{T}\right)$ or $\cos\left(\frac{2\pi b}{T}\right)$ as the target. Similarly, we use probes trained with $\sin\left(\frac{2\pi(a+b)}{T}\right)$ or $\cos\left(\frac{2\pi(a+b)}{T}\right)$ as the target to probe for structure in the `sum` variable.

Evaluating Fourier Probes. Figure 9 shows the R^2 scores for Fourier probes for every layer $l \in \{0, ..., 31\}$, and period $T \in \{2, ..., 100\}$ for both Llama-3.1-8B and several fine-tuned variants. We compute R^2 as the variance of the projected activations explained by the ideal ring after Procrustes alignment—measuring whether points lie on a circle at their correct phases. Across each setting, the same periods have high R^2 scores, but the LoRA fine-tuned models have stronger peaks (higher R^2 values) at the base-10 periods: $T \in \{2, 5, 10, 20, 50\}$ than the base model.

Figure 10 shows the R^2 scores of Fourier planes for layer 18 (i.e., row 18 of the heatmap in Figure 9).

Figures 11, 12, and 13 show a projection of the months data ($X_{\texttt{sum}\leq 96}$) onto the `input`, `offset`, and `sum` Fourier planes at layer 18 for base-10 periods. We include similar plots of the `sum` for layer 20: Figures 14, and layer 22: Figure 16.

Figure 15 shows the R^2 scores of Fourier planes for layer 22 (i.e., row 22 of the heatmap in Figure 9).

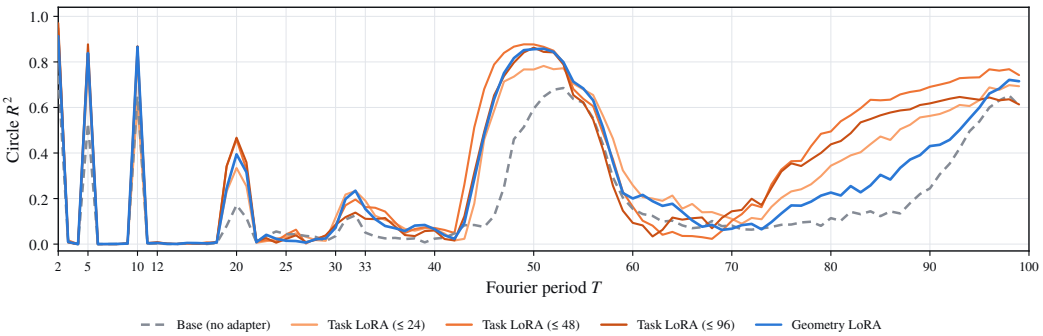


Figure 10: LoRA fine-tuning sharpens existing Fourier representations at Layer 18. We compute the R^2 value between Fourier planes and ideal circular representations at Layer 18 in Llama-3.1-8B. Task LoRAs (orange) and a geometry LoRA (blue) increase the R^2 of base-10 periods ($T \in \{2, 5, 10, 20, 50\}$) compared to the base model (shown in gray), leaving most other high-frequency periods largely unaffected.

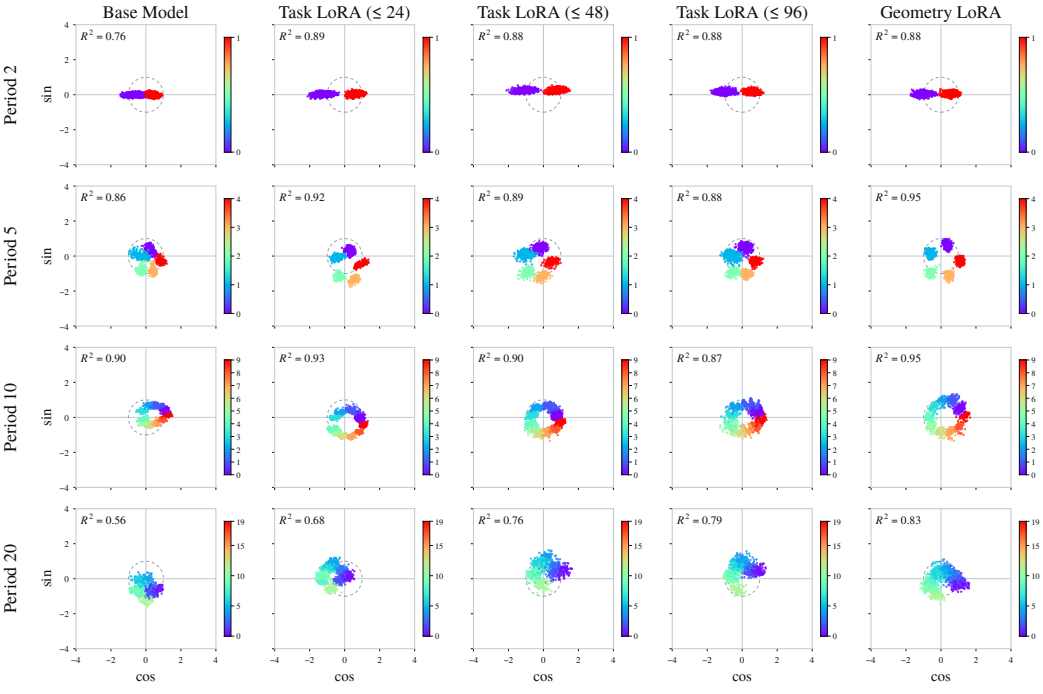


Figure 11: Projection of the months data onto Fourier planes of the `input` variable for periods $T = \{2, 5, 10, 20\}$ at layer 18. Each point corresponds to a prompt $x \in X_{\texttt{sum}\leq 96}$, and is colored by $\texttt{input}(x) \bmod T$. We compare projections from the base model (left column), and several LoRAs: Task LoRA ($\texttt{sum} \leq 24$), Task LoRA ($\texttt{sum} \leq 48$), Task LoRA ($\texttt{sum} \leq 96$), and a Geometry LoRA ($\texttt{sum} \leq 24$). Since `input` only has $|\mathcal{V}_c| = 12$ possible values, projecting it onto the period $T = 20$ plane forms an arc instead of a circle.

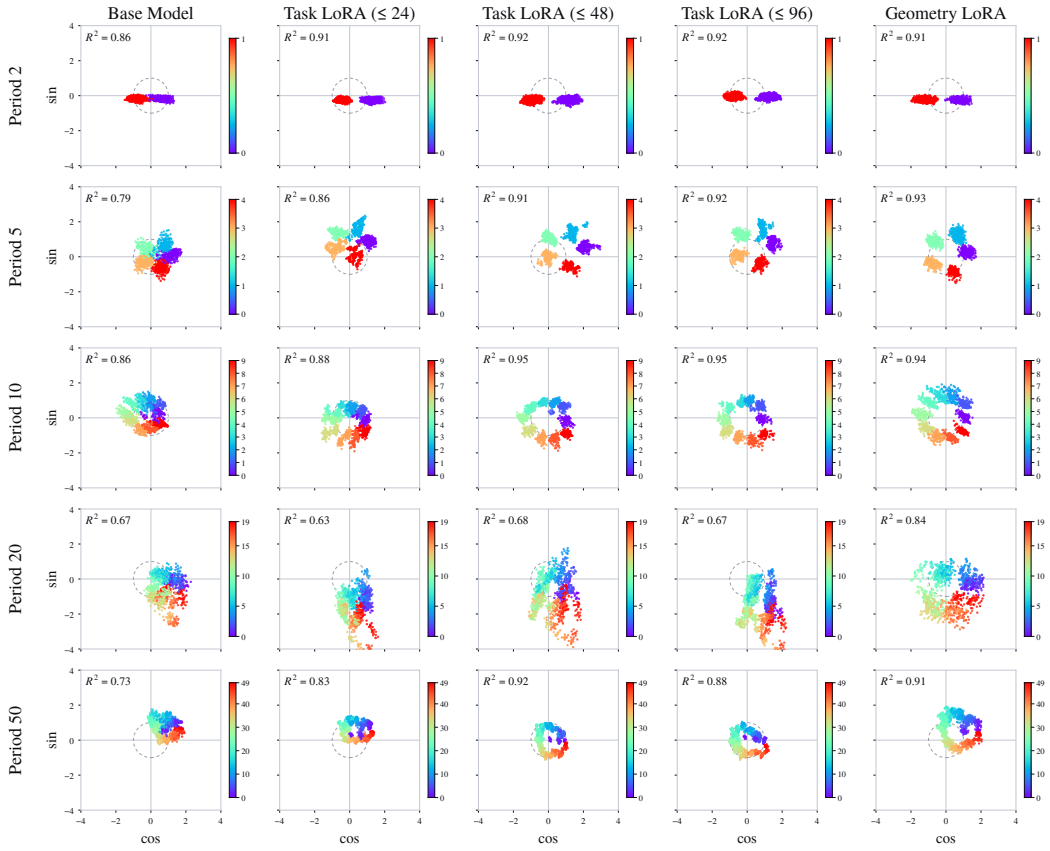


Figure 12: Projection of the months data onto Fourier planes of the `offset` variable for periods $T = \{2, 5, 10, 20, 50\}$ at layer 18. Each point corresponds to a prompt $x \in X_{\texttt{sum}\leq 96}$, and is colored by $\texttt{offset}(x) \bmod T$. We compare projections from the base model (left column), and several LoRAs: Task LoRA ($\texttt{sum} \leq 24$), Task LoRA ($\texttt{sum} \leq 48$), Task LoRA ($\texttt{sum} \leq 96$), and a Geometry LoRA ($\texttt{sum} \leq 24$).

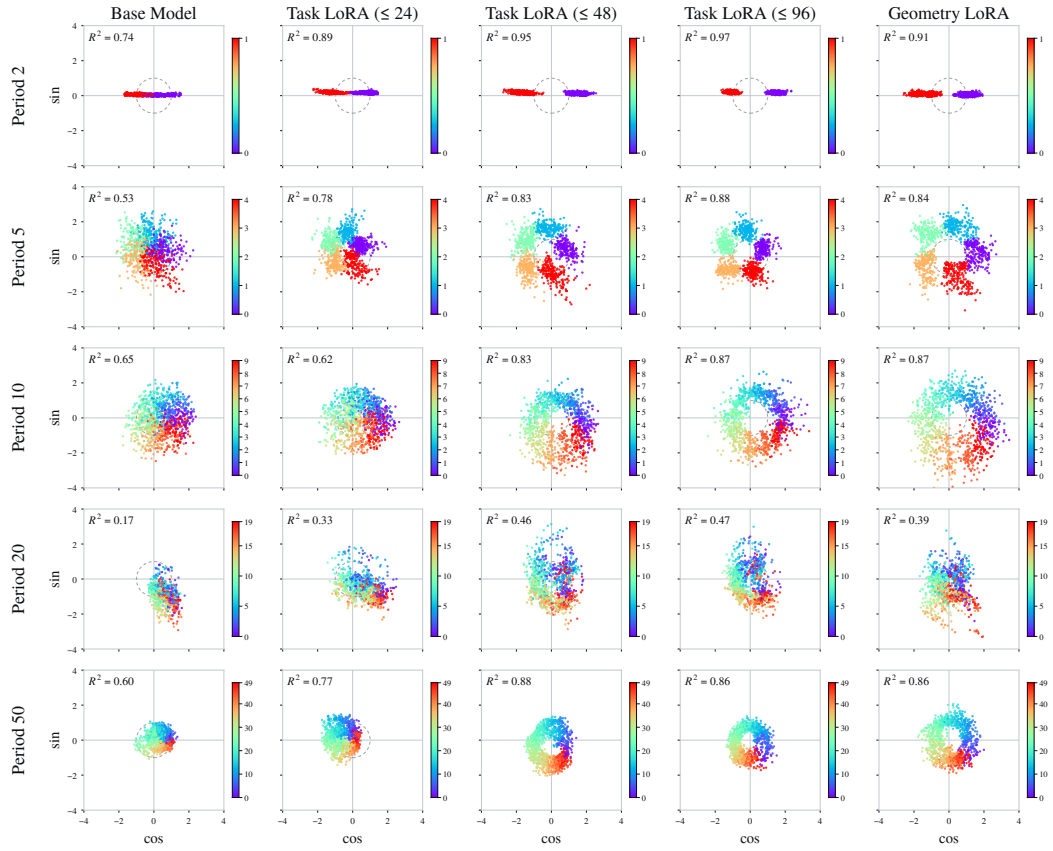


Figure 13: Projection of the months data onto Fourier planes of the sum variable for periods $T = \{2, 5, 10, 20, 50\}$ at layer 18. Each point corresponds to a prompt $x \in X_{\text{sum} \leq 96}$, and is colored by $\text{sum}(x) \bmod T$. We compare projections from the base model (left column), and several LoRAs: Task LoRA ($\text{sum} \leq 24$), Task LoRA ($\text{sum} \leq 48$), Task LoRA ($\text{sum} \leq 96$), and a Geometry LoRA ($\text{sum} \leq 24$).

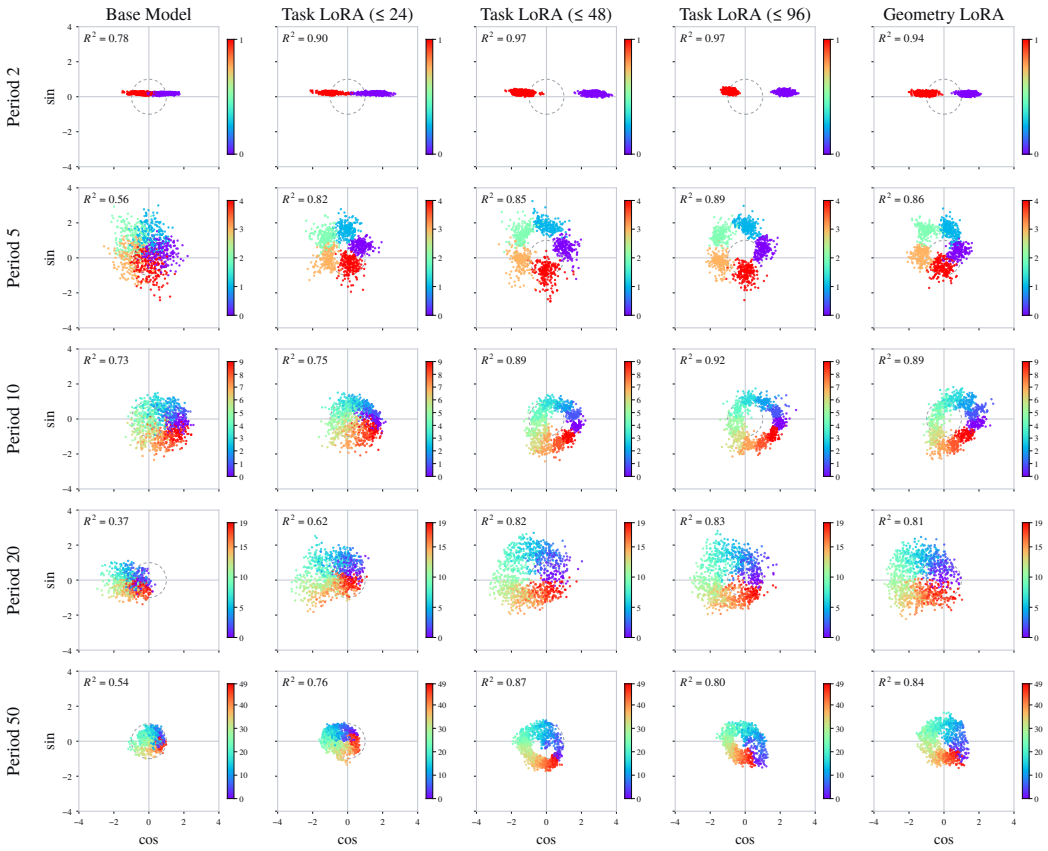


Figure 14: Projection of the months data onto Fourier planes of the sum variable for periods $T = \{2, 5, 10, 20, 50\}$ at layer 20. Each point corresponds to a prompt $x \in X_{\texttt{sum}\leq 96}$, and is colored by $\texttt{sum}(x) \bmod T$. We compare projections from the base model (left column), and several LoRAs: Task LoRA ($\texttt{sum} \leq 24$), Task LoRA ($\texttt{sum} \leq 48$), Task LoRA ($\texttt{sum} \leq 96$), and a Geometry LoRA ($\texttt{sum} \leq 24$).

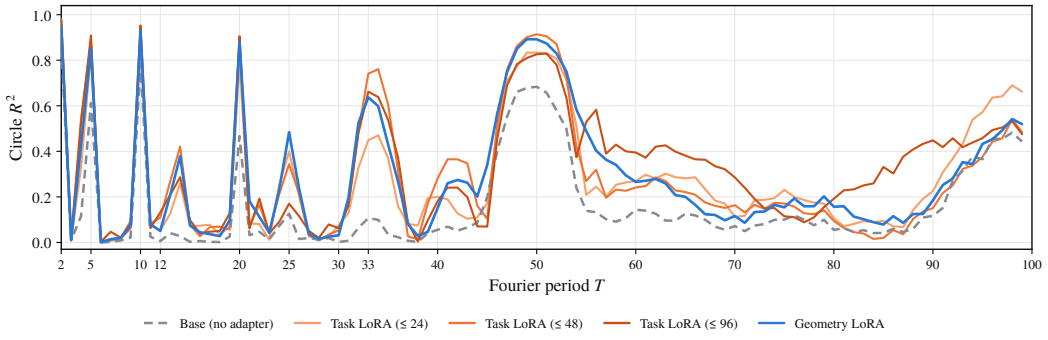


Figure 15: LoRA fine-tuning sharpens existing base-10 Fourier representations at Layer 22. We compute the R^2 value between Fourier planes for the sum and ideal circular representations at Layer 22 in Llama-3.1-8B. Task LoRAs (orange) and our Geometry LoRA (blue) increase the R^2 of base-10 periods ($T \in \{2, 5, 10, 20, 50\}$) compared to the base model (shown in gray), leaving most other periods largely unaffected. Interestingly, for this layer, additional periods also appear to sharpen which are not (strongly) present in the base model including $T \in \{14, 25, 33\}$.

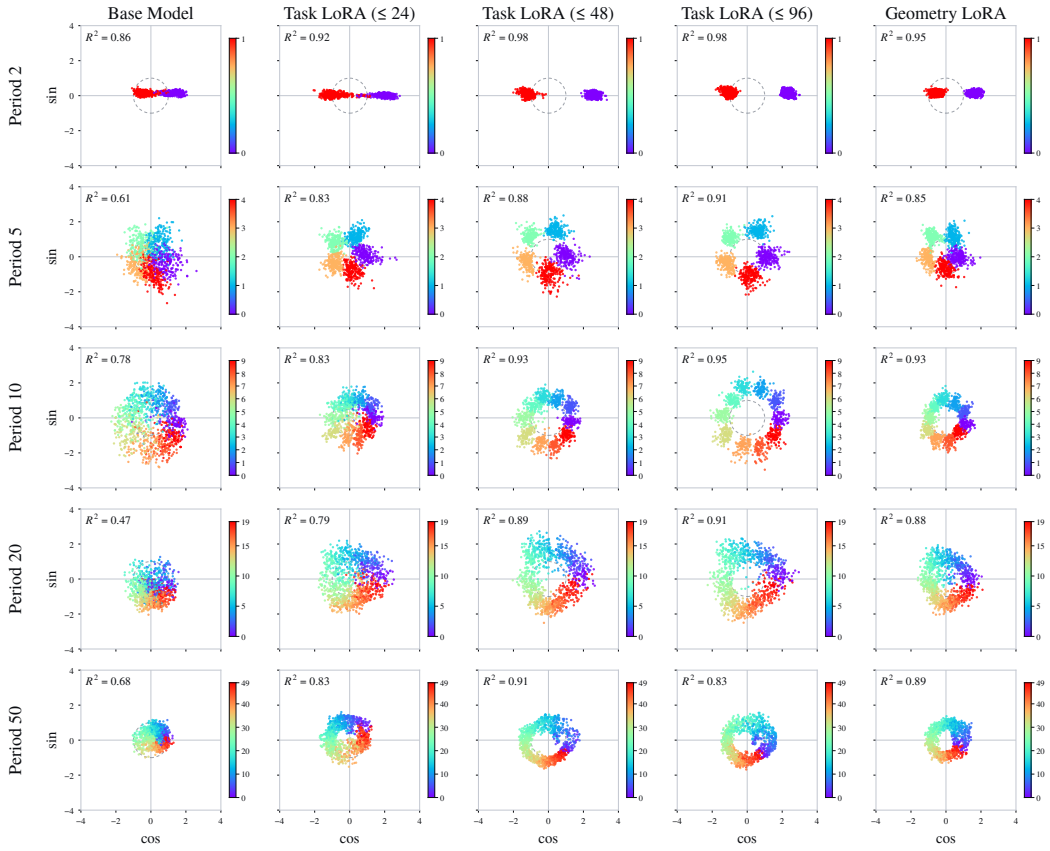


Figure 16: Projection of the months data onto Fourier planes of the sum variable for periods $T = \{2, 5, 10, 20, 50\}$ at layer 22. Each point corresponds to a prompt $x \in X_{\texttt{sum} \leq 96}$, and is colored by $\texttt{sum}(x) \bmod T$. We compare projections from the base model (left column), and several LoRAs: Task LoRA ($\texttt{sum} \leq 24$), Task LoRA ($\texttt{sum} \leq 48$), Task LoRA ($\texttt{sum} \leq 96$), and a Geometry LoRA ($\texttt{sum} \leq 24$).

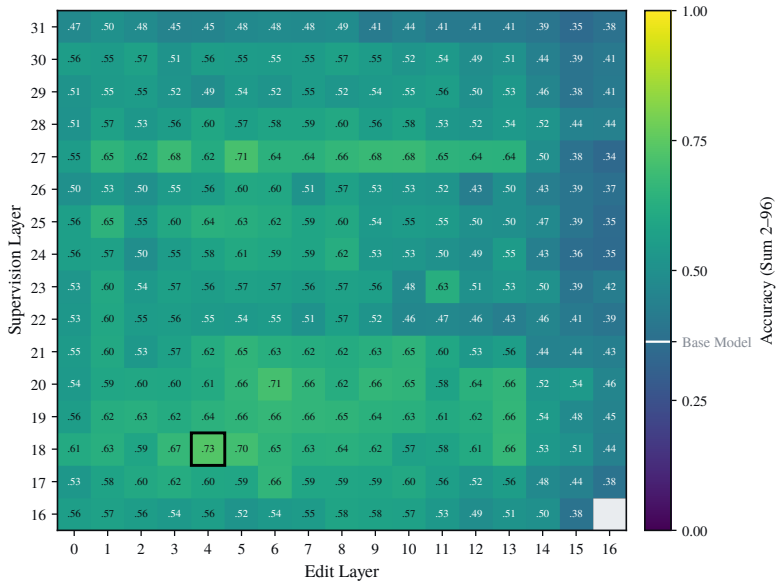


Figure 17: Comparing performance of geometry LoRAs trained and supervised at different layer combinations. The edit layer, where the LoRA is applied, ranges from $0-16$ and supervision (where the loss is computed) ranges from layer $16-31$. Each cell represents the accuracy of a rank-1 geometry LoRA trained on $X_{\texttt{sum}\leq 24}$ and scored over the full months dataset: $X_{\texttt{sum}\leq 96}$. The geometry LoRA configuration used in the main paper (edit layer 4, supervision layer 18, rank-1) is shown with a black box. Supervising before layer 16 or editing at layer 17 or later gives little to no improvement over the base model. The best edit layers are around layers $3-7$, while the best supervision layers are around layers $18-20$, with layer 27 also having strong effects.

A.3 Geometry LoRA Training Details

In this section, we provide additional details about how we train LoRAs with geometric supervision. Training a geometry LoRA requires making two choices: (1) where to supervise the geometry (i.e., via a projection $\mathbf{W}$), and (2) what the geometry should look like (i.e., an idealized target geometry $\mathbf{G}$). For the months task, all geometry LoRAs use a set of Fourier planes whose union behaves like a projection $\mathbf{W}$, and assume the target structure matches ideal Fourier plane geometry.

The geometry LoRA results reported in the main paper use a rank-1 LoRA trained at the MLP of layer 4 (i.e., the gate, up, and down projections), and supervised at the final-token residual stream at layer 18. We use $\alpha_r = 1.45$ for the initial procrustes alignment of the target centroids $\boldsymbol{\mu}_v$, a spread ratio target of $\tau = 1.25$, and $\lambda = 0.3$ to balance the two loss terms. The frame of each plane (the base model's mean centroid $\bar{\boldsymbol{\mu}}_c(\theta_0)$, radius r_0, and orientation $\mathbf{Q}_0$) is fit once using the base model's projected hidden states over $X_{\texttt{sum}\leq 24}$ and held fixed. We use the same optimizer settings as the task LoRA (§ A.1) and train for 200 gradient steps over the same training set used by the task LoRA. Averaged over three seeds, this geometry LoRA reaches 73.3% accuracy on the months task.

Sweeping over edit layer and supervision layer. In Figure 17, we show the results of geometry LoRAs trained and supervised at different layer combinations. The edit layer, where the LoRA update is applied ranges from $0-16$, and the supervision layers range from $16-31$. The geometry LoRA configuration used in the main paper (edit layer 4, supervision layer 18, rank-1) is shown with a black box. We find the best edit layers are in the range of $3-7$ while the best supervision layers are around $18-20$, aligning with the Feucht et al. (2026)'s localization of the sum computation.

LoRA Rank. In Figure 18, we sweep the LoRA rank over $\{1, 2, 4, 8, 16\}$ at every edit layer 0–16 with geometric supervision fixed at layer 18. We find that the best cell at each rank lies at edit layers 3–6, with ranks 1–4 achieving similar scores (e.g., at the layer-4 MLP they score 73.3%, 73.1%, and 71.1% overall). Because of this, we use rank-1 geometry LoRAs throughout.

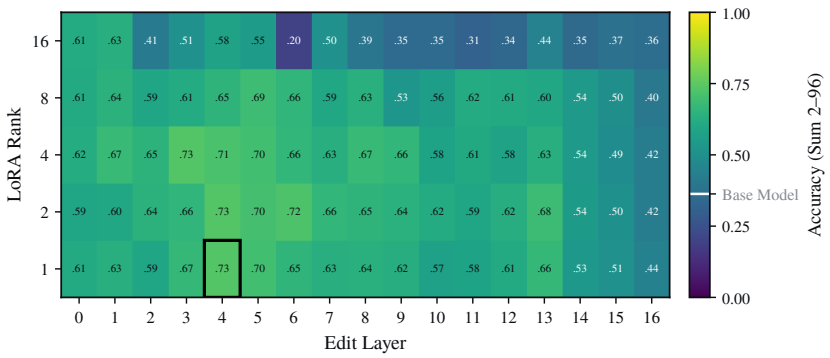


Figure 18: Comparing LoRA rank vs. edit layer. Months task performance of geometry LoRAs trained at different edit layers ($0-16$) with different ranks ($\{1,2,4,8,16\}$) for a fixed supervision layer ($l = 18$). We report performance on the full dataset $X_{\texttt{sum}\leq 96}$. We find that training a geometry LoRA on the MLP within layers $3-6$ has the strongest effect, while rank has little effect (ranks $1-4$ achieve similar performance for a given edit layer). The geometry LoRA configuration used in the main paper (edit layer 4, supervision layer 18, rank-1) is shown with a black box.

Geometry LoRA on a single layer vs. multilayer. In Table 2, we compare the effect of training a LoRA on just a single layer to training LoRAs on multiple layers. For the geometry LoRAs, we fix the supervision layer to layer 18, and report either a single LoRA trained at the layer 4 MLP, or trained at all layers (MLP, and MLP + Attention) up to the supervision layer (i.e., $0-17$). Training geometry LoRAs with multiple edit layers follows the opposite trend compared to task LoRAs trained at multiple layers. Training on all layers can improve performance on near-range sums ($\texttt{sum} \in [25-48]$) compared to the single layer LoRA, but performance drops on the far-range sums ($\texttt{sum} \in [49-96]$) by about the same amount.

Table 2: Comparing single vs. multi-layer LoRA training. Each LoRA is rank-1 and trained on $X_{\texttt{sum}\leq 24}$ with only geometric supervision at layer 18. We report the accuracy in three evaluation bands. The first band is in-range: $\texttt{sum} \in [2, 24]$, the other two are held-out ranges: $\texttt{sum} \in [25, 48]$, and $\texttt{sum} \in [49, 96]$. Scores are mean $\pm$ standard deviation over 3 seeds. Multi-layer LoRAs use a learning rate of 5×10^{-4}.

				Accuracy (%)		
Objective	Location	Layers	# Params.	$\texttt{sum} \in [2, 24]$	$\texttt{sum} \in [25, 48]$	$\texttt{sum} \in [49, 96]$
–	Base Model	–	0	79.0%	22.9%	27.4%
Geometry	MLP	4	55K	$96.8 \pm 0.7\%$	$66.9 \pm 7.1\%$	$67.9 \pm 3.8\%$
Geometry	MLP	0–17	0.99M	$98.9 \pm 0.3\%$	$77.1 \pm 0.3\%$	$51.2 \pm 1.2\%$
Geometry	MLP + Attn	0–17	1.47M	$98.6 \pm 0.0\%$	$78.0 \pm 0.9\%$	$48.8 \pm 0.6\%$

A.4 Ablations of the Geometric Objective

Per-prompt vs. per-value supervision. The geometry loss in Equation 3 aligns empirical per-`value` centroids $\hat{\boldsymbol{\mu}}_v(\theta)$ to their targets $\boldsymbol{\mu}_v$. A natural alternative is to supervise each prompt directly, aligning each projected activation $\mathbf{W}\mathbf{h}_x(\theta)$ to its target centroid. This can be achieved by replacing each centroid $\hat{\boldsymbol{\mu}}_v(\theta)$ in Equation 3 with individual activations $\mathbf{W}\mathbf{h}_x(\theta)$, and averaging over all prompts in the training set $X_{\texttt{sum}\leq 24}$:

$$\mathcal{L}_{\text{prompt}} = \frac{1}{|X|} \sum_{v \in \mathcal{V}_c} \sum_{x \in X_v} \frac{1}{r_0^2} \left\| \mathbf{W}\mathbf{h}_x(\theta) - \boldsymbol{\mu}_v \right\|_2^2 \tag{12}$$

where r_0^2 is the same as in Section 2.

We compare per-prompt and per-value supervision for training geometry LoRAs and find that both objectives can achieve good performance on the months task. The best per-prompt geometry LoRA achieves 69.6% accuracy, while the best per-value geometry LoRA achieves 73.4% accuracy (see Table 3). We find that per-prompt supervision is more dependent on the spread term ($\mathcal{L}_{\text{spread}}$; Equation 4) than per-value supervision. Ablating the spread term (setting $\lambda = 0$ in Equation 2) drops the performance of per-prompt supervision by $\sim$14.6%, from 69.6% to 45.0%. while only dropping performance of per-value supervision by $\sim$2.2%, from 73.4% to 71.2% (see Table 3, λ=0).

One reason for this dependence on the spread term might be because the per-prompt loss comes with an implicit regularization on the spread that drives each value toward a single point. We can write $\mathbf{W}\mathbf{h}_x(\theta) - \boldsymbol{\mu}_v = \big(\mathbf{W}\mathbf{h}_x(\theta) - \hat{\boldsymbol{\mu}}_v(\theta)\big) + \big(\hat{\boldsymbol{\mu}}_v(\theta) - \boldsymbol{\mu}_v\big)$, and because deviations from a centroid sum to zero, the cross term vanishes and the per-prompt loss can be written as the sum of a centroid-matching term and within-value spread term:

$$\mathcal{L}_{\text{prompt}} = \sum_{v \in \mathcal{V}_c} \frac{|X_v|}{|X|} \frac{1}{r_0^2} \big\|\hat{\boldsymbol{\mu}}_v(\theta) - \boldsymbol{\mu}_v\big\|_2^2 \; + \; \frac{\sigma_c(\theta)^2}{r_0^2}, \tag{13}$$

where $\sigma_c(\theta)$ is the within-value spread of Equation 5. The first term is $\mathcal{L}_{\text{mean}}$ with each value weighted by its number of prompts, and the second is minimized at $\sigma_c(\theta) = 0$, causing the collapse to a single point. The spread loss term $\mathcal{L}_{\text{spread}}$ in Equation 4 instead anchors $\sigma_c(\theta)$ at a ratio of the base model spread ($\tau \cdot \sigma_c(\theta_0)$), which can effectively "rescue" the spread-collapsing term in $\mathcal{L}_{\text{prompt}}$. We see this in Figure 19, which projects activations at layer 18 onto the `offset` Fourier planes and shows that points collapse to target centroids when $\lambda = 0$, but recover for non-zero values of λ.

Table 3: Per-prompt vs. per-value supervision, with and without the spread term, $\mathcal{L}_{\text{spread}}$ (Eq. 4). All rows use the standard geometry LoRA settings (rank 1, edit-layer 4, supervised at layer 18) and differ only in the loss. Accuracy is the mean $\pm$ standard deviation over 3 seeds. We report accuracy on the months task, as well as in-range (`sum` $\in [2, 24]$) and held-out ranges (`sum` $\in [25, 48]$, `sum` $\in [49, 96]$). Per-value with $\lambda = 0.3$ is the geometry LoRA reported in the main text (marked with *).

Supervision	λ	`sum` $\in [2, 96]$	`sum` $\in [2, 24]$	`sum` $\in [25, 48]$	`sum` $\in [49, 96]$
Per-value	0	$71.2 \pm 2.8\%$	$92.5 \pm 2.1\%$	$63.2 \pm 1.6\%$	$67.4 \pm 5.3\%$
Per-value*	0.3*	$73.3 \pm 4.1\%$	$96.8 \pm 0.7\%$	$66.9 \pm 7.1\%$	$67.9 \pm 3.8\%$
Per-value	1	$73.4 \pm 1.0\%$	$96.7 \pm 1.3\%$	$66.7 \pm 1.2\%$	$68.3 \pm 2.8\%$
Per-value	3	$69.8 \pm 2.0\%$	$93.8 \pm 3.3\%$	$62.5 \pm 2.1\%$	$64.8 \pm 1.5\%$
Per-prompt	0	$45.0 \pm 3.8\%$	$65.4 \pm 7.7\%$	$29.3 \pm 3.9\%$	$45.4 \pm 2.9\%$
Per-prompt	0.3	$68.7 \pm 1.9\%$	$94.3 \pm 2.7\%$	$62.6 \pm 0.2\%$	$62.4 \pm 4.6\%$
Per-prompt	1	$69.3 \pm 1.8\%$	$94.0 \pm 0.5\%$	$60.6 \pm 1.1\%$	$64.6 \pm 3.9\%$
Per-prompt	3	$69.6 \pm 2.7\%$	$94.6 \pm 2.7\%$	$62.4 \pm 3.8\%$	$64.1 \pm 2.3\%$

Sensitivity to α_r and τ. The geometry loss (Equation 2) has two hyperparameters: (1) the radius α_r, which determines how far the target centroids are from the center relative to the base model f_{θ_0}, and (2) the spread ratio τ, which determines the within-value spread relative to the base model. We sweep $\alpha_r \in \{0.8, 1.0, 1.2, 1.45, 1.7, 2.0, 2.5, 3.0\}$ and $\tau \in \{0.5, 0.8, 1.0, 1.25, 1.5, 2.0\}$ using the geometry LoRA training settings described in S A.3, and report results in Figure 20.

A.4.1 Fourier Planes as a Supervision Site

In this section, we evaluate two design choices: (1) whether supervising geometry on a subspace spanned by trained Fourier probes is a meaningful choice and (2) the effect of supervising an increasing number of Fourier planes.

Are Fourier planes a meaningful supervision site? We measure whether our choice of the Fourier planes as the supervision site $\mathbf{W}$ is meaningful by training a geometry LoRA in the same way on a random subspace of the same dimension. To supervise our geometry LoRA we use eight Fourier planes ($\{c \in$ `input-month`, `offset`$\} \times T \in \{2, 5, 10, 20\}$), thus for our random control we sample eight random orthonormal Gaussian planes and place ideal targets on a ring the same way as we do for the Fourier probe planes (see Equation 9). We use the same training settings as the standard geometry LoRA, only having substituted our Fourier plane construction of $\mathbf{W}$ for a random projection to a subspace of the same dimension.

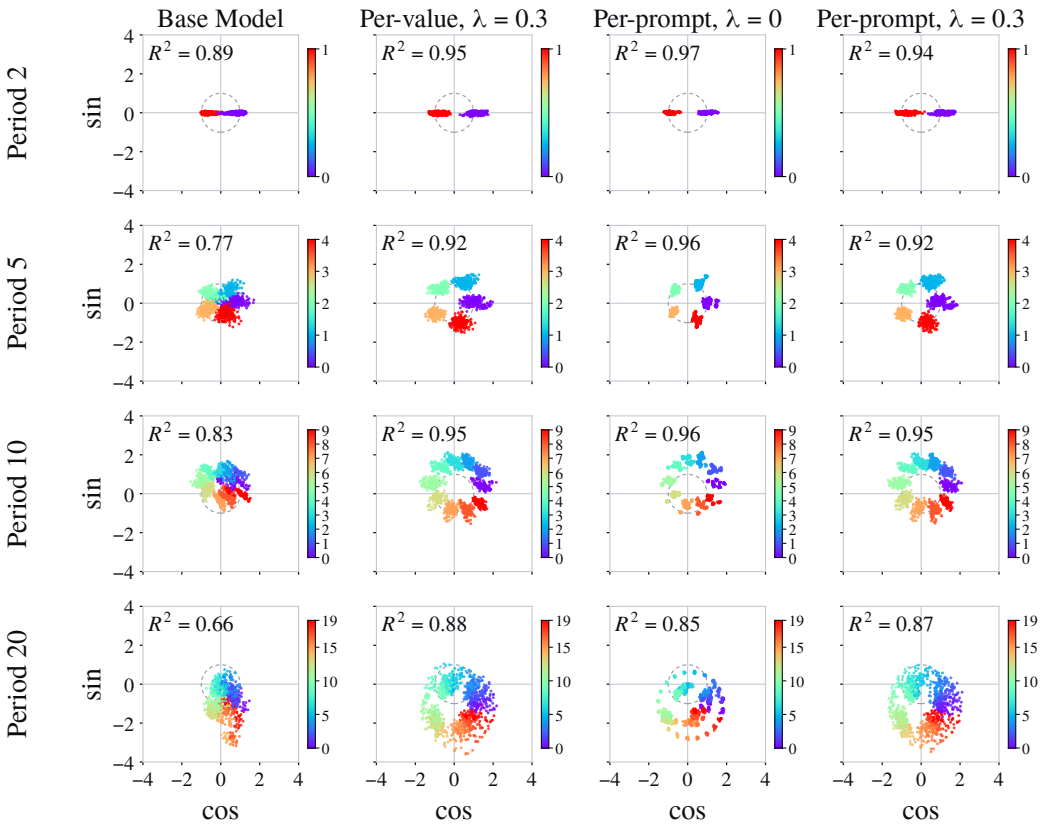


Figure 19: Layer-18 projections of all data ($X_{\texttt{sum}} \leq 96$) onto the `offset` planes for the base model, a per-value geometry LoRA, and geometry LoRAs trained with the per-prompt loss without ($\lambda = 0$) and with ($\lambda = 0.3$) the spread term, colored by `offset` $\bmod T$. Per-prompt supervision without the spread term collapses training data to each residue (third column), but introducing the spread term restores the within-class spread (fourth column).

We find that a random-subspace geometry LoRA achieves an overall accuracy of 28.4% on the months task, compared to 73.3% when using the Fourier planes, and 36.3% for the base model. Similarly, we find that training a geometry LoRA within the Fourier planes, but randomly permuting the value-to-centroid assignment also hurts performance, dropping from 36.3% to 10.2%.

Supervising too many Fourier planes hurts performance. We measure the effect that training geometry LoRAs with an increasing number of supervised periods (e.g., $\mathcal{T}=\{2\}, \mathcal{T}=\{2,5\}, \mathcal{T}=\{2,5,10\}, ..., \mathcal{T}=\{2, ..., 100\}$) has on performance (see Figure 21). We fix the supervision layer at layer $l^* = 18$, and add periods to the supervision set $\mathcal{T}$ by first including the base-10 Fourier features (2, 5, 10, 20, 50) in order, and then selecting others based on their R^2 value (see Figure 10). We find that using an increasing number of periods for geometric supervision hurts performance, and that four periods: $\mathcal{T}=\{2, 5, 10, 20\}$ is the best supervision set. Surprisingly, supervising only a single Fourier plane ($\mathcal{T}=\{2\}$) already achieves around 63.0% accuracy, while using planes of four periods gets 73.3% accuracy.

A.5 Evaluating Off-Target Effects of Fine-tuning

In this section, we evaluate the end-to-end and geometry LoRAs on several external tasks to measure their effect on the model's general capabilities. These include: FineWeb (Penedo et al., 2024), WikiText-103 (Merity et al., 2017), MMLU (Hendrycks et al., 2021), and GSM8K (Cobbe et al., 2021). On FineWeb, we measure bits per byte (bpb) over 600 documents. On WikiText-103, we measure perplexity (ppl). For MMLU we evaluate on the full test set using 5-shot prompting and

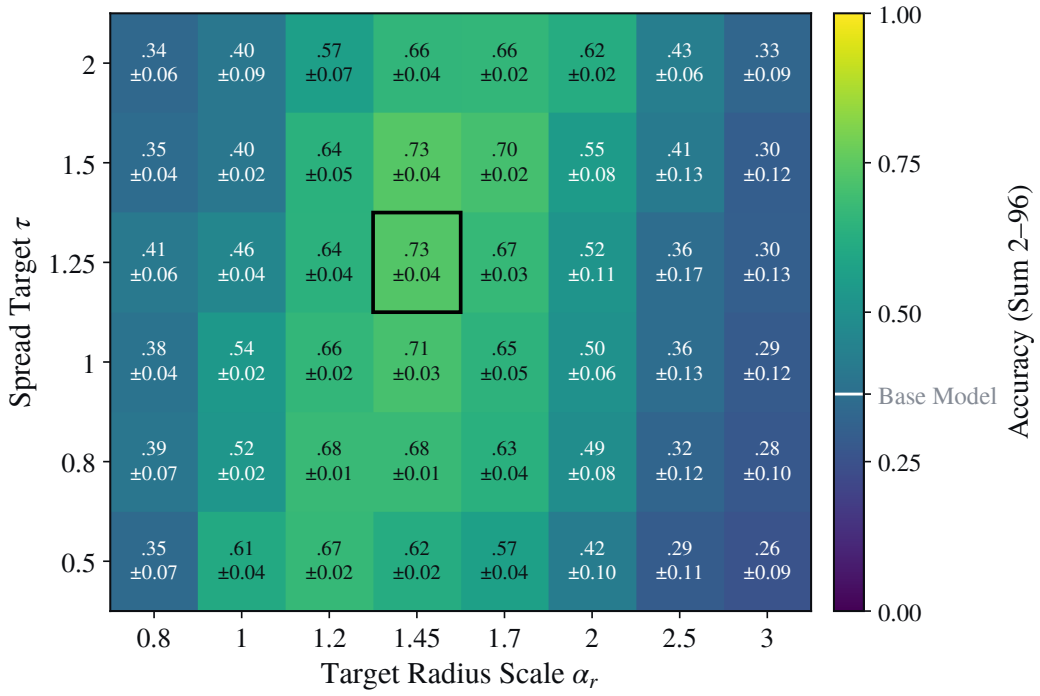


Figure 20: Sensitivity of the geometry LoRA to the radius α_r and the spread target ratio τ. Each cell shows the months-task-performance of a rank-1 geometry LoRA trained at layer 4, supervised at layer 18, with $\lambda = 0.3$. Accuracy is reported as a mean over 3 seeds ($\pm$ standard deviation). The black box denotes the geometry LoRA used in the main paper ($\alpha_r = 1.45$, $\tau = 1.25$). We find that performance depends mainly on α_r, while τ has a smaller, but still relevant effect.

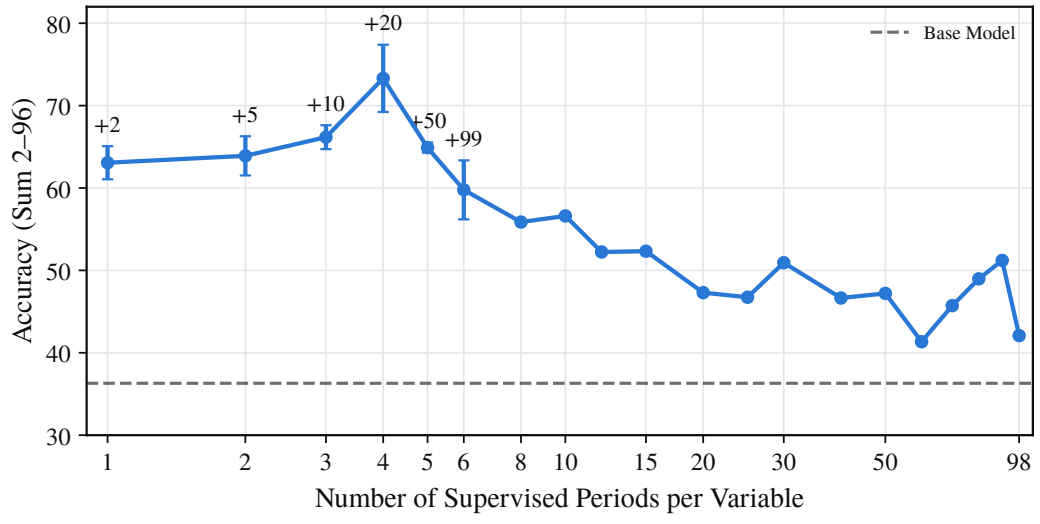


Figure 21: Supervising on many Fourier planes hurts performance. We measure the effect on performance of increasing the number of supervised Fourier planes for geometry LoRAs. Periods are chosen first from $\{2, 5, 10, 20, 50\}$ in order, and then by their probe's R^2 value. The x-axis is on a log scale, as we test supervising Fourier planes from up to 98 periods. We find that supervising more than six periods hurts performance compared to using just a single period ($T = 2$), which surprisingly achieves 63.0% accuracy. The best configuration uses four periods: $\mathcal{T} = \{2, 5, 10, 20\}$, which achieves 73.3%. The base model's performance is plotted as a dashed line (36.3%)

report the accuracy (%). For GSM8K, we evaluate on the full test set using 8-shot chain-of-thought with greedy decoding and compute the exact match accuracy on the final output number. We find that the geometry LoRA and task LoRAs have little to no effect on general model performance as measured by these other tasks (see Table 4).

Table 4: Off-task effects of fine-tuning on the months task. We evaluate Llama-3.1-8B and LoRA fine-tuned variants on several LM benchmark tasks: FineWeb bpb, WikiText-103 ppl, MMLU, and GSM8K to measure the effect of fine-tuning on external tasks. Rows reporting LoRA results are means over 3 seeds.

Model	Months acc. (%) ↑	FineWeb bpb ↓	WikiText ppl ↓	MMLU (%) ↑	GSM8K (%) ↑
Base Model	36.3	0.709	7.75	65.4	52.5
Geometry LoRA (≤ 24)	73.3	0.709	7.75	65.5	52.0
Task LoRA (≤ 24)	71.9	0.710	7.78	65.0	53.5
Oracle Task LoRA (≤ 96)	100.0	0.709	7.75	65.1	51.9

A.6 Comparing Geometric Supervision to Other Baselines

In this section, we report additional baselines on the months task: few-shot prompting (ICL), standard finetuning, and Constructive Circuit Amplification (CCA) (Prakash et al., 2025), a targeted fine-tuning method that locates and then trains task-specific circuits.

For each baseline/control, we use $X_{\texttt{sum}\leq 24}$ as our training set. For few-shot prompting, we test using $\{4, 8, 32\}$ in-context examples. CCA has two settings, where a learned masked is applied ("w/ mask") or not ("w/o mask").

Table 5: Additional methods evaluated on the months task: few-shot prompting, full fine-tuning, and CCA. Few-shot examples are drawn from $\texttt{sum} \leq 24$ and excluded from scoring when present in the prompt. Full fine-tuning uses the task loss with the learning rate swept over 10^{-6}–10^{-3} (best rate, final step, one seed). LoRA and CCA rows are means over 3 seeds.

Method	# params	$\texttt{sum} \in [2, 96]$	$[2, 24]$	$[25, 48]$	$[49, 96]$
Base model	–	36.3%	79.0%	22.9%	27.4%
Few-shot, 4 examples	–	44.4%	95.1%	44.8%	25.7%
Few-shot, 8 examples	–	41.2%	77.7%	35.4%	30.7%
Few-shot, 32 examples	–	40.6%	83.7%	33.3%	28.5%
CCA, w/ mask (Prakash et al., 2025)	296M	55.5%	100.0%	69.1%	32.5%
CCA, w/o mask (Prakash et al., 2025)	8.03B	62.7%	100.0%	81.2%	39.8%
Full fine-tuning, all layers	8.03B	63.2%	100.0%	78.5%	42.2%
Full fine-tuning, layers 0–15	3.49B	62.7%	100.0%	76.7%	42.0%
Full fine-tuning, layers 16–31	3.49B	49.8%	100.0%	57.3%	27.8%
Task LoRA, rank 1, layer 4	55K	54.3%	100.0%	68.2%	30.7%
Task LoRA, rank 8, layer 3	442K	71.9%	100.0%	78.0%	58.6%
Geometry LoRA, rank 1, layer 4	55K	73.3%	96.8%	66.9%	67.9%

B Linebreaking Task: Experimental Details

B.1 Linebreaking Dataset Examples

In this section, we provide a few examples of wrapped FineWeb (Penedo et al., 2024) documents from our linebreaking dataset. See § 4 for dataset construction details. Figure 22 shows an example of a document being wrapped both at m=20 characters per line, and m=40 characters per line. Figure23 shows an example of a document wrapped to m=80 characters.

Wrapped to m=20 characters

```
During the spring\n
and summer of 1946\n
two groups of WWII\n
Veterans held\n
meetings to discuss\n
creating new taxi\n
companies. One group\n
called themselves\n
Tuxedo Cab, the\n
other Radio Cab. . . .
```

Wrapped to m=40 characters

```
During the spring and summer of 1946 two\n
groups of WWII Veterans held meetings to\n
discuss creating new taxi companies. One\n
group called themselves Tuxedo Cab, the\n
other Radio Cab. Each group appeared\n
before the Portland City Council to\n
request taxi company operating permits.\n
City Council told them to leave, join\n
forces and come back as one company.\n
They got together, appeared before . . .
```

Figure 22: An example document from the linebreaking dataset. The same FineWeb passage (a Radio Cab company history) is wrapped to width $m = 20$ (left) and $m = 40$ (right): a newline (\n) is inserted at the last word boundary that keeps each line to at most m characters.

Wrapped to m=80 characters

```
September is a confusing time for seasonal purists. I'm not saying I'm one, but\n
I do tend to measure the progress of the seasons by what I eat and at the middle\n
point of this month I am always conflicted. There are still a great many fruits\n
available that I associate with summer (berries, grapes, figs and melons to name\n
a few) but what happens in September is we see an overlap into the fall harvest\n
season. While the growers of summer are still churning out some of the best\n
fruit of the season, apple and pear growers throughout the U.S. are entering\n
into the game and the competition at farmers' markets and grocery stores for . . .
```

Figure 23: A second example from the linebreaking dataset (a FineWeb passage on seasonal eating), wrapped to width $m = 80$. As before, a newline (\n) is inserted at the last word boundary that keeps each line to at most m characters.

B.2 Training Task LoRAs for Linebreaking

In this section, we provide additional details about how we train task LoRAs for the linebreaking task. All task LoRAs are trained with a standard next-token cross-entropy loss at every token position. We train a rank-1 LoRA on the attention and MLP projections of layers $0{-}5$, and the LoRA update is applied at every token position. We use the AdamW optimizer (Loshchilov & Hutter, 2019) with a learning rate of 1×10^{-4}, gradient-norm clipping at 1.0, and a constant learning rate schedule, and we train for 400 steps with a batch size of 4 over $1{,}600$ FineWeb documents, each wrapped to a random width $m \in [20, 150]$. The base model is Llama-3.1-8B in `bf16` with weights kept frozen, while the LoRA parameters are kept in `fp32`. We use this same LoRA placement—rank, layers, optimizer, and training data—for both the task and geometry LoRAs, so that the comparison between them isolates the loss function: the two differ only in whether the objective is the output-level cross-entropy or geometric supervision (§B.3).

B.3 Training Geometry LoRAs with Helix-Supervision

In this section, we provide additional details about how we train geometry LoRAs for the linebreaking task. As with the months task, we must choose (1) where to supervise the geometry (i.e., a projection $\mathbf{W}$), and (2) what the geometry should look like (i.e., an idealized target geometry $\mathbf{G}$). For the linebreaking task, the projection we use $\mathbf{W} \in \mathbb{R}^{3\times d}$ is a helix frame: two Fourier probes $\mathbf{w}_{\cos}, \mathbf{w}_{\sin}$ trained to match $\cos(2\pi\,\texttt{character-count}(x_t)/T)$ and $\sin(2\pi\,\texttt{character-count}(x_t)/T)$ at period T, and the linear `character-count` probe $\mathbf{w}_{\texttt{count}}$, and we choose the target structure to be the ideal helix geometry. We fit all three probes on the base model using tokens from FineWeb documents wrapped to a single width $m = 150$ over counts $20{-}150$, with ridge regression.

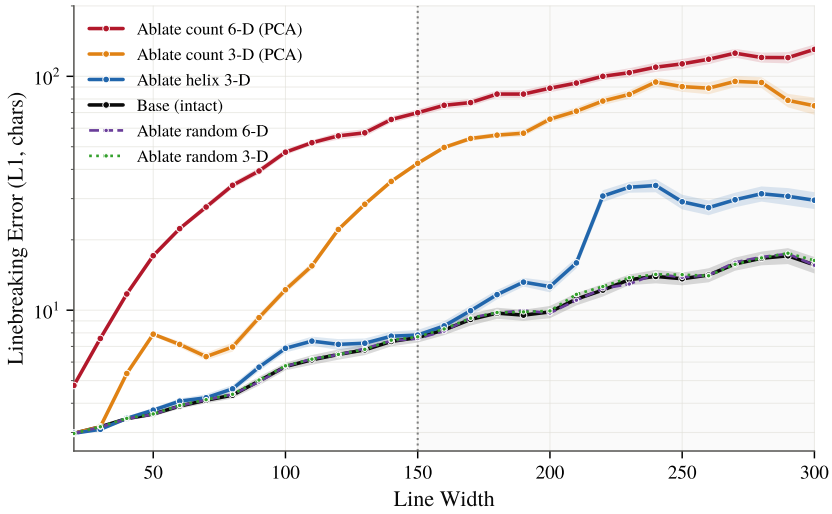


Figure 24: Linebreaking error as a function of line width under zero-ablation of the `character-count` subspace at layer 6. Ablating the top-6 or top-3 PCA directions of the per-count means sharply raises the error, growing with width, while ablating random subspaces has no effect. Ablating the helix frame (the Fourier plane plus the count axis) has little effect in-range, but increases linebreaking error as line width increases.

The idealized target places each `character-count` value v on the ideal helix point $\mathbf{u}_v = (\cos(2\pi v/T),\ \sin(2\pi v/T),\ v/T)$, which we align to the base model's per-`character-count` centroids (computed over counts $1{-}150$) via orthogonal Procrustes once before training and keep fixed throughout (Equation 9). The geometry LoRA results reported in the main paper use a rank-1 LoRA trained on the attention and MLP projections of layers $0{-}5$. We use a radius of $\alpha_r = 1.2$, a spread ratio target of $\tau = 0.5$, and $\lambda = 1.0$ to balance the two loss terms. We use the same optimizer settings, training data, and number of steps as the task LoRA (§B.2), and supervise at each token at layer $l^* = 6$.

Choosing the period T. We set $T = 155$, the period at which the base model's `character-count` representation completes a single rotation in the Fourier plane. We determine this by taking the first two PCs of the per-count means (i.e. the two rotating components in Figure 5) and compute the phase of each count as $\theta_v = \mathrm{atan2}(\mathrm{PC}_2, \mathrm{PC}_1)$. We unwrap the phase across counts and fit a line to predict the phase by least squares: $\theta_v = \omega v + b$. The period is then computed as $T = 2\pi/\omega$, which yields $T \approx 154.77$ characters.

B.4 Character-Count Manifold Experiment Details

In this section, we provide more details about experiments we run to characterize the `character-count` manifold.

PCA zero-ablation experiments. We test whether the mean `character-count` subspace is causally used by the model by zero-ablating it at layer 6 (i.e., projecting activations off the top-k PCA directions of the per-count means) and measuring the effect on linebreaking error. As a control, we apply the same ablation to a random k-dimensional subspace. We find that ablating the count subspace increases linebreaking error by a large margin, and increasingly so at wider widths, while the random control leaves the error at the base model's level (Figure 24). Ablating only the top-3 directions already inflates the error substantially (e.g., from 7.6 to 42.5 characters at width $m = 150$), and ablating the top-6 does so at every width.

Alignment to an idealized helix. To measure how closely each model's representation follows an idealized helix, we align a period-T helix ($\mathbf{u}_v = (\cos(2\pi v/T),\ \sin(2\pi v/T),\ v/T)$, $T \approx 155$) to the layer-6 hidden states of all tokens—aligning each token to the ideal position for its `character-count`—via orthogonal Procrustes, and report the fraction of variance the aligned

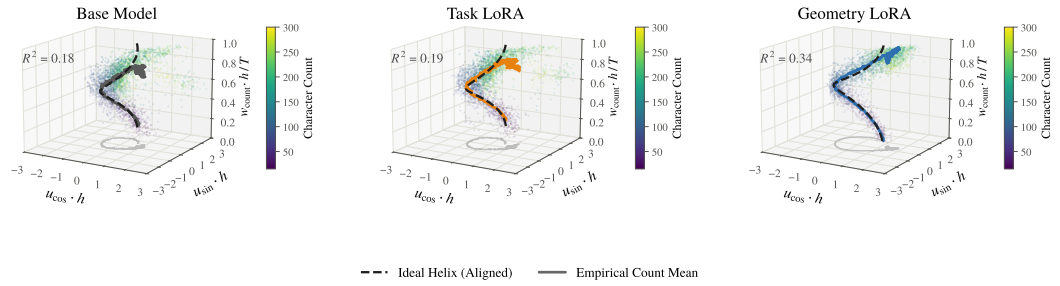


Figure 25: The layer-6 `character-count` manifold under the **(a)** base model, **(b)** task LoRA, and **(c)** geometry LoRA. We measure the fraction of variance an idealized helix captures over all tokens (R^2). Plotted points are individual token hidden states projected onto a fixed helix (§ 4.2) and colored by `character-count` $\in [20, 300]$. The solid curve in each plot is the empirical per-`character-count` mean and the dashed curve is the idealized helix. In-range ($m \leq 150$), the activations follow the helix, but for the extended held-out range ($m \in [151, 300]$), the alignment drops significantly for each model. The base model achieves R^2=0.18, the task LoRA achieves R^2=0.19, and the geometry LoRA achieves R^2=0.34. The drop in alignment comes as the helix does not rotate a second time within the 3 supervised dimensions.

helix captures (R^2). Because it is computed over all tokens rather than the per-count means, this metric folds in the within-count spread, making it a combined measure of the manifold's shape and tightness. Over the in-range counts (15−150), the base model already follows the helix well ($R^2 = 0.69$), the task LoRA sharpens it slightly ($R^2 = 0.73$), and the geometry LoRA aligns most tightly ($R^2 = 0.93$; Figure 6). Extended to the full count range (15−300), the alignment of all three models drops sharply (0.18, 0.19, and 0.34 respectively; Figure 25) as the representation departs from the helix far out of distribution.