How Do Language Models Manage Uncertainty?: A Case Study of the Number Game

Anonymous authors
Paper under double-blind review

Abstract

Language models must navigate ambiguity to successfully complete sentences or perform tasks; however, we have little mechanistic understanding of how they represent or handle such uncertainty. To understand this better, we study how Llama-3.1 (8B) plays the Number Game, a concept learning task in which participants view a set of numbers that conform to an unknown concept, and must determine which other numbers conform. Llama's behavior, like humans', appears Bayesian; we search for the mechanisms underlying this seemingly normative behavior. We first find that Llama's representational geometry constrains the concepts that it considers as hypotheses, allowing it to consider only a specific range of possible hypotheses. We next find that Llama has internal circuitry that identifies common concepts in its input, and downweights poorly supported ones. Finally, we find that Llama's increased confidence upon seeing more inputs stems largely from an attention mechanism that simply tracks the *number* of examples seen. We use these insights to design a new model that better predicts Llama's behavior, including on new concept types. In sum, we offer a detailed account of how Llama manages its uncertainty and what mechanisms underlie this capacity.

1 Introduction

Modern language models (LMs) often act under uncertainty, to predict the next token of syntactically ambiguous sentences, or fulfill user requests with ambiguous intents. Prior work in mechanistic interpretability has mostly focused on reverse-engineering deterministic tasks, such as arithmetic (Kantamneni & Tegmark, 2025; Nikankin et al., 2025; Feucht et al., 2026). This work has amassed a great deal of insight into how models encode and manipulate representations—e.g., performing addition on individual numbers—but it does not address how models manage uncertainty over representations. How, for example, does a model encode uncertainty between two numbers, or more generally, between two hypotheses about a variable of interest?

Cognitive scientists have studied analogous questions in humans using tasks like the *Number Game* (Tenenbaum, 1999a), depicted in Figure 1, in which participants are presented with a sequence of numbers and asked to predict the next number. This requires maintaining and updating uncertainty over the *underlying rule* that generated the data. Tenenbaum (1999a) found that a straightforward Bayesian model predicted human judgments well. Like much work in Bayesian cognitive science (Griffiths et al. 2010), this analysis remained at Marr's (Marr, 2010) *computational level*, leaving open the mechanisms by which peoples' behavior matches the normative Bayesian predictions.

In this work, we analyze Llama 3.1 (8B)'s behavior on the Number Game. Like in the earlier human studies, we find that a simple Bayesian model predicts Llama's behavior quite well. But unlike in the human case, here we are able to dig much deeper into *how* precisely Llama performs this task, crucially including how it manages its uncertainty over a space of latent hypotheses.

A first observation concerns the space of hypotheses that are entertained. Drawing on a wealth of recent interpretability research, we find that representational geometry plays a major role. Specifically, many hypotheses are encoded using Fourier features, the very same ones that underlie Llama's arithmetic ability (Feucht et al., 2026). This effectively functions as a representational bias: Hypotheses that might be simple for other kinds of systems—e.g., *multiples of 7*—appear not even to be considered. We identify the exact circuits used to detect common patterns in lists of numbers.

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.

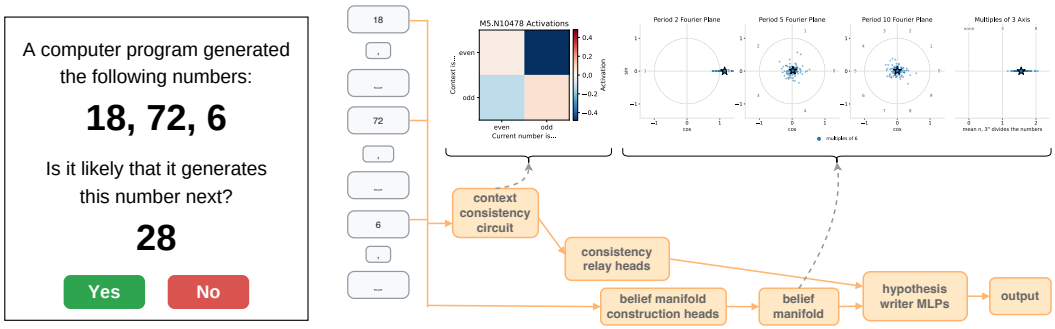


Figure 1: **Left**: An instance of the Number Game. Players view a set X of numbers that matches an unknown concept, and determine if a query number y matches, too. Players must reason about the correct underlying concept, such as *multiples of 6* vs. *2*, which may be underspecified. **Right**: Llama's circuit for detecting multiples of 6. Llama's belief about the hypothesis is represented on various Fourier planes (Section 3), constructed by a set of averaging heads (Section 4). A context consistency circuit verifies that all input elements have the same parity (Section 4). The manifold and circuit output both influence the MLPs that construct the output.

A second key observation concerns how hypotheses are progressively updated. A normative (Bayesian) method would update hypotheses in response to (the number of) data points that weigh against, or in favor of, that hypothesis. We find that Llama, by contrast, implements an attention-based mechanism that treats *amount of evidence* more indiscriminately, without tracking precisely how the evidence weighs with respect to alternative hypotheses.

Following these and other insights, we craft a revised model, grounded in the mechanistic details that we uncover. The revised model predicts Llama behavior on the Number Game even better than the first-pass Bayesian model, and generalizes well to novel settings. Gaining a better understanding of how Llama performs in the Number Game, we begin to see more generally how language models deal with uncertainty. The present study thereby contributes both to mechanistic interpretability—going beyond the more common deterministic settings—and the cognitive sciences, providing a case study for the mechanisms underlying inference under uncertainty in a neural architecture.

2 The Number Game

Players in the Number Game (Tenenbaum, 1999a;b; Tenenbaum & Griffiths, 2001) are given a collection of integers X sampled from an unknown concept set $C \subseteq \mathbb{N}$; they must determine if a new number y is also in C (Figure 1). C is typically a rule such as *multiples of* n or *numbers in the interval* $[m, n]$; the space of potential concepts is not known to the participant. In Tenenbaum & Griffiths's proposed model of behavior in the Number Game, the likelihood $p(x|h) = \mathbb{1}[x \in h]/|h|$ follows the *size principle* and shrinks as $|h|$ grows (as examples are sampled uniformly from h). Therefore, smaller hypotheses (e.g., *multiples of 6*) are judged more likely than larger ones (e.g., *even numbers*) given data that support both.

The Number Game for Language Models. We adapt the Number Game for LMs, by having them predict the next token in a sequence. We give a base LM its BOS token and N numbers from C, each followed by a comma and space, without context, e.g. "[BOS]6, 28, 40, 18, " and record its $p(y|X)$. We consider all numbers 1-999 as potential targets in the Number Game. Our concept space adapts that of Bigelow & Piantadosi (2016), including: multiples of 2-20, numbers that start with 1-9 or end with 0-9, odd numbers, and intervals from m to n, for $m \leq n$. We mainly study Llama-3.1 8B (Llama Team, 2024), which is frequently studied in interpretability and tokenizes all numbers 1-999 as a single token, facilitating our experiments; see Appendix O for Llama-3.1 70B.

To assess Llama's behavior on the Number Game, we construct inputs X by sampling N numbers with replacement from each C. For each C, and each N from 1 to 30, we generate 100 inputs, or all unique inputs if there are fewer than 100. We measure performance via the probability Llama gives to numbers in the ground-truth hypothesis C: $p(Y \in C|X) = \sum_{y \in C} p(y \mid X)$.

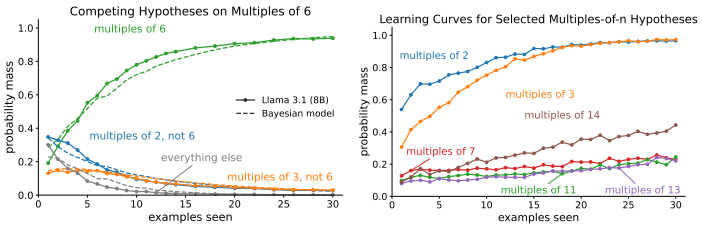


Figure 2: **Left**: Llama's behavior on inputs taken from *multiples-of-6* (solid line). It quickly assigns more probability to multiples of 6 than to multiples of 2 or 3; this preference grows with the number of examples. The Bayesian model (dashed) replicates this behavior relatively well, though it assigns slightly too much probability to multiples of 2. **Right**: Llama's performance on various *multiples-of-n* hypotheses. Llama learns *multiples of 2* or *3* easily but fails to learn the others.

We find that Llama successfully plays the Number Game. In Figure 2 (left), given inputs drawn from *multiples of 6*, Llama assigns most of its probability to multiples of 6 after only 3 examples. In doing so, it follows the size principle, dispreferring the larger but valid concept *multiples of 2*. However, Llama does not learn all hypotheses: e.g., Figure 2 (right) shows that it never learns *multiples of 7*, assigning them very low probability (see Appendix B for full behavioral results). Our results contrast with past work, which found that Llama failed zero-shot on the Number Game (Padmanabhan et al., 2025; Bazigaran & Sohn, 2025); this is because these studies used the original Yes/No Number Game format, instead of having Llama predict the next number.

Bayesian Modeling of LM Behavior. Tenenbaum & Griffiths proposed a model of Bayesian inference over a latent hypothesis space $\mathcal{H}$:

$$p(y \in C|X) = \sum_{h \in \mathcal{H}} p(y \in C|h)p(h|X) \propto \sum_{h \in \mathcal{H}} \mathbb{1}[y \in h]p(X|h)p(h) \tag{1}$$

where each $h \in \mathcal{H}$ is a potential hypothesis (i.e. a concept set C) the observer considers. We adapt this model to next-token prediction, and add a power-law term to account for LMs' slow evidence accumulation (Bigelow et al., 2026; Wurgaft et al., 2025):

$$p_B(y|X) \propto \sum_{h \in \mathcal{H}: y \in h} \frac{\mathbb{1}[X \subseteq h]}{|h|} \exp\left(\gamma_h |X|^{1-\alpha} + \beta_h\right). \tag{2}$$

This model learns a log-prior β_h and log-likelihood term γ_h for each hypothesis h, as well as a power-law parameter α. We fit this model using KL divergence on Llama's population-level outputs, i.e. its average outputs for a given (C, N), as Llama's per-output logits are noisy. The Bayesian model fits Llama's population-level outputs reasonably well, with a KL divergence of 0.074. Qualitatively, Figure 2 (left) shows that the Bayesian model's predictions track Llama's, differing only on short inputs (See Appendix C for extended analysis and model details).

However, a behavioral match does not imply that Equation (2) reflects Llama's internal computations; we thus study the mechanisms underlying Llama's Number Game performance. We focus on the following questions, each matching a color-coded component of Equation (2): (1) What hypothesis space $\mathcal{H}$ does Llama consider, and how are the hypotheses represented? (2) How does Llama deal with ambiguity—does it eliminate hypotheses that are incompatible with the evidence? (3) How and how quickly does Llama accumulate evidence?

3 Representing Beliefs About Concepts

In this section, we characterize the representational geometry of Llama's beliefs about hypotheses. We show that this both constrains the hypotheses Llama can represent, explaining its failure to learn *multiples of 7*, and supports new hypotheses missing from the Bayesian model's hypothesis space $\mathcal{H}$. We use causal interventions to show these beliefs are responsible for Llama's behavior.

We locate Llama's beliefs about the current hypothesis using distributed alignment search (DAS; Geiger et al., 2024). DAS learns a low-rank subspace of model activations such that transplanting the subspace's contents from one input X' into the forward pass on another input X transfers Llama's

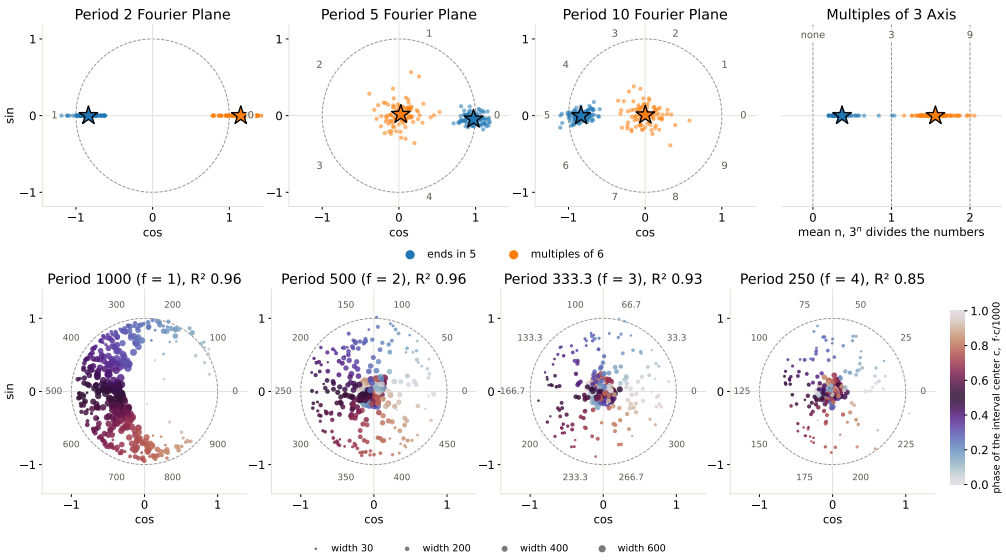


Figure 3: **Top.** Layer 26 Fourier-plane representations of *ends-in-5* (blue) and *multiples-of-6* (orange): dots are activations, and stars indicate their centroid. Each period-T Fourier plane encodes the hypothesis mod T. The *multiples-of-3* axis records whether a number is a multiple of 3 or 9. **Bottom.** Layer 26 Fourier-plane representations of intervals. Each dot is the representation of one interval h: its color represents h's center mod T (the period); its size represents the h's width. Intervals are encoded using many long-period Fourier features. Wider intervals lie closer to the center.

predictions accordingly: a subspace that supports such swaps could carry Llama's beliefs. We learn these subspaces at the final space token of each input, where Llama predicts the next number. Subspaces become learnable at layer 15, with performance improving up to layer 23; the belief subspace has roughly 32 dimensions (see Appendix D for details on this and other experiments in this section). With this subspace in hand, we can now examine how beliefs are represented within it.

Past work on arithmetic in LMs suggests that numbers are often encoded with Fourier features, i.e., as periodic functions of their values, rather than their raw values (Nanda et al., 2023; Zhong et al., 2023; Gould et al., 2024). We thus use *Fourier probes*, which probe for representations of a number x via $\sin$ or $\cos(2\pi x/T)$, with T as the period (Levy & Geva, 2025); Such sine-cosine pairs constitute a circular encoding of x mod T: plotting one probe's output against the other places x values on a circle, with angular position determined by remainder mod T (Feucht et al., 2026).

***Multiples-of-n* and *Ends-in-n*.** We train Fourier probes on Llama's representations of inputs X drawn from *multiples-of-n* and *ends-in-n*. We probe Llama's DAS subspaces (at layers 18-31) for the mean Fourier feature of the numbers in X, at periods $T = 2, 5, 10, 20, 50, 100$. These probes achieve high accuracy from layer 18 onward, but only for $T = 2, 5, 10$; probing $T \geq 20$ fails.

Plotting the probes' outputs shows that Llama represents the members of *multiples-of-n* and *ends-in-n* hypotheses mod T. For example, *ends-in-5* is represented as 0 mod 2, 0 mod 5, and 5 mod 10 (Figure 3, top); indeed, these properties are true of all numbers ending in 5. By contrast, *multiples-of-6* is 0 mod 2 but has no clear value on the period-5 and 10 planes, as multiples of 6 can take on many values mod 5 or 10. The representational geometry of the Fourier planes can thus represent not only the output of deterministic operations like addition (Feucht et al., 2026), but also uncertainty: when an input takes on many values mod T, its average period-T Fourier feature lands at the center of the period-T Fourier plane, indicating no strong belief.

Notably, these features cannot represent all *multiples-of-n* hypotheses: they would leave *multiples of 6* encoded identically to *multiples of 2* or *4*. To resolve this, we successfully train probes to take in *multiples-of-n* representations from the DAS subspace, and predict the multiplicity of 2 as a factor of *n*; we do the same for 3. Llama has dedicated directions, but not Fourier planes, for these features.

Together, these manifolds support all of the *multiples-of-n* and *ends-in-n* hypotheses that Llama learns. Some, like *multiples of 2*, have a dedicated Fourier plane. Others, like *multiples of 6*, are

represented as the conjunction of two rules (*multiples of 2* and *3*), but lack their own manifold. Those hypotheses that are not representable, like *multiples of 7, 11, 13, 14, 17* and *19*, are not learned.

Intervals and *Starts-with-n*. We observe that Llama can learn interval hypotheses, but these are not representable with short-period Fourier features. Based on past work that found Gaussian distributions are encoded by LMs via their mean and variance (Sarfati et al., 2026), we hypothesize that Llama encodes intervals via their center and width. As such, we train Fourier probes to predict the mean Fourier feature of an input X drawn from an interval, at periods $T = 1000, 500, 333, 250, 200$, i.e. the first 5 fundamental frequencies of a period-1000 signal. We also probe directly for X's width.

We find that Llama does encode intervals using their center and width (Figure 3, bottom). Plotting the interval inputs with respect to each Fourier plane shows that each interval lies at the mean of its period-T Fourier features. Strikingly, wider intervals lie closer to the origin of each plane, i.e., they have lower amplitude, especially at shorter periods. This is because averaging over wide windows of periodic functions naturally produces lower values; no explicit width computation is needed to produce this effect. We note Llama nonetheless computes width, which is both decodable across layers and causally relevant; see Appendix E. Finally, the *starts-with-n* manifold is fundamentally similar to that of width-100 intervals centered at $n \times 100 + 50$; only one axis separates them from the corresponding interval manifold embedding; see Appendix D for details.

Causal Intervention. These manifolds are causally implicated in Llama's behavior. Given an input consisting of e.g. *multiples-of-6*, we can cause Llama to output numbers ending in 5 by manipulating the relevant Fourier planes. Crucially, we understand these manifolds in such depth that we can analytically compute the embedding of any given hypothesis h on our manifolds. If we compute h's embedding and patch it into Llama, we can cause it to output h. See experiments in Appendix E.

Discussion. Llama's feature geometry reflects what it can and cannot learn in-context. It can represent and learn $x = 12\ mod\ 15$ as the conjunction of *multiples of 3* and $x = 2\ mod\ 5$, but cannot learn a simple rule like *multiples of 7*. Llama's hypothesis space also contains hypotheses not discussed here (Appendix B.1) but it does not contain or represent *multiples of 7* (Figure 2, right).

These experiments reveal differences between Llama and the Bayesian model. First, our Bayesian model omitted hypotheses of the form $= n$ mod 5 for $n \neq 0$ from its hypothesis space, but Llama can represent and learn them (Section 6). Our new model must include these in its $\mathcal{H}$. Second, while in the original Bayesian model, all hypotheses compete for probability mass in the computation of $p(h|X)$, Llama's beliefs in hypotheses do not appear to be normalized. While there is competition between hypotheses on the same manifold—simultaneous belief in *even numbers* and *odd numbers* is impossible, as their representations lie on opposite sides of the period-2 Fourier manifold—many of these manifolds are orthogonal to each other, so there is no inherent, geometric competition (Appendix F). Activating two hypotheses at once yields conjunction, not competition, or disjunction.

4 Representing and Acting on Uncertainty

The prior section sheds light on Llama's conceptual representations. Next, we explore how Llama uses these for inference under uncertainty. We make two claims: first, Llama constructs its initial belief manifolds by averaging the input's Fourier features. This allows its geometry to naturally encode ambiguity that exists in its inputs (Figure 4, left). Second, Llama has additional, non-averaging mechanisms that allow it to modulate its certainty in a given belief: the logits of a member of h shrink rapidly as the input becomes less consistent with h. However, it does not eliminate hypotheses after just one inconsistent input, like the Bayesian model's indicator function $\mathbb{1}[X \subseteq h]$.

Constructing Beliefs. If Llama's beliefs are formed by averaging per-number features, its belief manifolds should reflect that fact. Consider, for example, an ambiguous input X ($|X| = 20$), of which some fraction f end in 4, and the rest end in 7. We project the resulting activations onto layer-16 post-attention period-10 manifold while varying f, and find that their position clearly tracks f (Figure 4, center). When all numbers end in 4 ($f = 1$), the representation sits at the $x = 4 \bmod 10$ position; as f falls, the representation moves towards the $x = 7 \bmod 10$ position. In other words, this movement corresponds to averaging of per-number Fourier features. This representation encodes uncertainty: when $f = 0.5$, its angle matches *ends-in-5/6* but it sits near the plane's origin, indicating low certainty. Surprisingly, even in this case, Llama still outputs numbers that end in 4 or

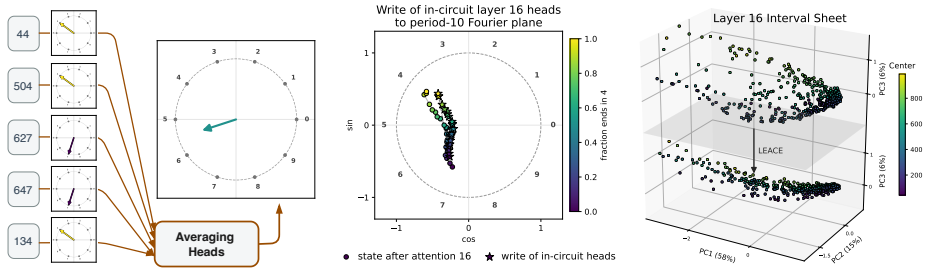


Figure 4: **Left**: Llama's attention heads construct the belief manifold as an average of Fourier features. **Center**: Llama's layer-16 post-attention period-10 Fourier plane, given mixed *ends-in-4/7* input. As the fraction of *ends-in-4* numbers drops, the representations move along the chord from *ends-in-4* to *ends-in-7*. The stars show the output of Llama's in-circuit layer-16 attention heads, which are similar. **Right**: Using LEACE to remove Fourier features from the number-position value vectors of upstream in-circuit attention heads destroys the structure of layer 16's interval manifold.

7. This is because Llama represents both the fundamental frequency of the period-10 Fourier plane and its higher harmonics, where it is easier to represent *ends-in-4/7* ambiguity (Appendix K).

Next, by using circuit analysis, we find heads that perform averaging and construct these manifolds. For each manifold identified in Section 3, we use Matryoshka mask learning (Arora et al., 2026) to find sparse and faithful attention head circuits, retaining a median of 14/1024 heads; these preserve 90% of Llama's Number Game performance when all heads outside of the circuit are mean-ablated. See Figure 1 for a circuit visualization, and Appendix H for details and each manifold's circuit.

In each manifold's circuit, we find heads that attend from space positions to number positions, and have a roughly uniform distribution; they plausibly perform weighted averaging. We show that these heads compute a weighted average of the number features via causal intervention. First, we probe each input number's representation for the relevant period-T Fourier features, as well as features like *multiplicity of 2* or *multiple of 3*; these are easily decodable. Then, for a given head, we use LEACE (Belrose et al., 2023) to remove all linearly extractable information about the relevant features from the head's value vectors at the number positions. We compare the effect of this ablation on Llama's $p(Y \in C|X)$ to that of mean ablating all number-position value vectors; they are similar for almost all heads. Visualizing this operation's effects shows that it flattens the manifold (Figure 4, right).

These heads also write to manifolds. The output of the layer 16 averaging heads, projected onto the period-10 Fourier manifold from earlier (Figure 4, center) tracks the whole manifold, indicating that they are writing to the manifold, building it up. Finally, we take the gradient of relevant directions (e.g. the $x = 1 \mod 5$ direction), let it flow through the attention heads, and use this to probe numbers for that property, with high accuracy. We note that a small set of heads fails these tests; we will soon see that they play another role. For details on experiments in this section, see Appendix I.

Acting on Beliefs. While Llama represents ambiguous inputs as an average of input Fourier features, its write to a hypothesis h is not linear in the fraction of inputs matching h. To show this, we sample inputs X_f, where f is the fraction of the input drawn from h, and the rest is drawn uniformly at random. We record Llama's logit difference between numbers in and outside of h (Figure 5, left), and find that for every h, the logit difference is at or below zero until $f \geq 0.4$, at which point the difference grows super-linearly in f. This entails a steep drop-off as inconsistencies with h appear. A similar phenomenon is visible in Llama's Fourier manifolds as we vary f (Figure 5, center).

The mechanism responsible for this seemingly exponential growth is not purely based on averaging; we show this using circuit analysis. We find attention circuits that preserve model behavior when heads' number-position value vectors are patched with values from the same number in an $X_{0.5}$ context. That is, we hold a number's identity (used for averaging) constant, but manipulate the context in which it sits. The top heads that we uncover in these circuits are not the averaging heads from earlier, but rather those heads that failed our averaging tests. These heads transmit the output of circuits computing *list-consistency*, which we discover using neuron circuit analysis. Figure 5 (right) shows one example: A5.H22 computes the average parity of preceding numbers, and a downstream neuron fires if the current number's parity agrees; see Appendix J for details. We do not claim a

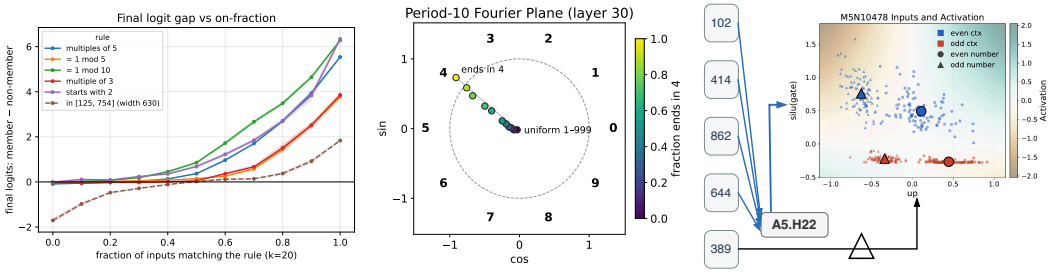


Figure 5: **Left**: Llama's logit difference between members and non-members of a hypothesis h vs. the fraction f of the input that is in h. The write grows quickly with f, past a threshold. **Center**: The norm of X_f, seen on the period-10 Fourier plane seems to grow exponentially with f. **Right**: A list parity consistency circuit in Llama. A5.H22 gathers the parity of numbers from the context, while neuron M5.N10478 multiplies the context's parity (encoded in its gate) by the current number's parity (in its up projection), to get its activation, which is positive only if they match; this contributes to the logit increase seen at left. The plot shows M5.N10478's activations plotted against their gate projection and up projection, when varying the parity of the context and current number.

full understanding of how these circuits work, or how they lead to Llama's exponential-in-f logits; however, they are, based on our circuits, causally relevant for this behavior.

Discussion. We find evidence that Llama computes an input's initial Fourier clock embedding as an average of per-number Fourier features; the resulting representation directly reflects ambiguity in the input. However, Llama's ultimate logits for a hypothesis h are not linear in the fraction f of examples matching h. Instead, they grow exponentially with f, staying near zero until $f = 0.4$ and then growing rapidly; this is supported in part by context-consistency circuits. This contrasts with our original Bayesian model, which eliminates hypotheses upon observing any incompatible data, but matches other noise-tolerant Bayesian models, and human judgments (Goodman et al., 2008). Based on this, our new model should be noise-tolerant, and avoid a strict indicator function.

5 Evidence Accumulation

Llama constructs its beliefs by averaging input numbers' features, and computing their internal consistency. While such operations are invariant to the number of examples seen, Llama's behavior is not: its MLPs write more strongly as it sees more examples compatible with a hypothesis (Figure 6, left)—even if the new evidence does not eliminate other hypotheses. This is also predicted by the Bayesian model, via its power-law scaling term $|X|^{1-\alpha}$. In this section, we identify the mechanism by which Llama's confidence increases over time and argue that it scales near-hyperbolically.

What drives Llama's behavior as X grows in length? Its attention heads must be responsible, since Llama encodes positional information via RoPE (Su et al., 2024), there is no other source of information about input length. We thus search for a new attention head circuit, using a new ablation. Concretely, let X be a length-15 input. We split X into 3 inputs, each containing a length-5 block of X's numbers: $X = [X_{1:5}, X_{6:10}, X_{11:15}]$; we store heads' outputs on these new inputs. To ablate a head at a given position in X, we replace its output with its matching length-5 output. We thus change Llama's perceived input length but maintain the actual length-15 input and its token identities. We find 95% faithful circuits, somewhat larger than the previous circuits (94 heads; Appendix L). Many (80%) of the heads that compute weighted averages of features are located at the space position of this circuit. We hypothesize that, as the number of non-BOS tokens in an input increases, the total attention mass to non-BOS tokens increases as well. The weighted average computed by these heads thus increases in strength because the sum of weights grows, boosting Llama's output. Figure 6 shows that these heads do attend more to non-BOS tokens as input length increases.

This is not, however, the only mechanism. We compare the effects of three interventions on each space-position head: 1) patch only the values of each head, while keeping the attention fixed; 2) patch only the attention pattern of each head, keeping its values fixed; 3) patch the head's output (as in the circuit). After each intervention, we record the change in $p(Y \in C|X)$. Since a head's output is the attention-weighted sum of its values, (1), (2), and their interaction fully explain (3). If attention-spreading dominates, the effects of (2) should be near those of (3). This is not the case. For

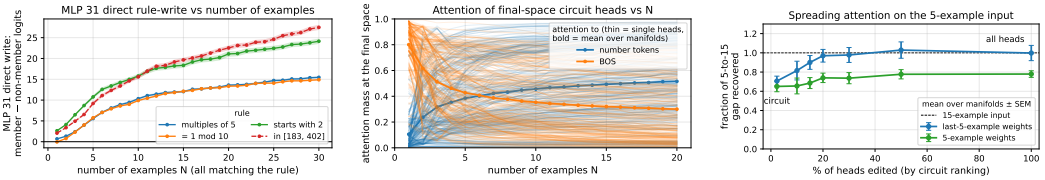


Figure 6: **Left**: The logit difference of the writes of Llama's MLP 31 towards various hypotheses h, given N examples drawn from h. The logit difference grows with N, even when new evidence does not directly eliminate hypotheses. **Center**: Attention given to non-BOS tokens by heads in evidence accumulation circuits grows with the length N of the input; attention to BOS drops. **Right**: Percentage of Llama's $p(Y \in C|X)$ on a length-15 input recovered by manipulating attention patterns on a length-5 input. We compare two different methods for redistributing attention to non-BOS tokens.

some heads, the effects of (2) and (3) are similar; for others, (1) is more harmful than (2), so these heads' values also change as Llama sees more examples. The attention spreading of number-position heads must drive this value change, but the precise mechanism is unclear.

If Llama's behavior as $|X|$ grows is driven by changes in attention distributions, then manipulating its attention should increase its $p(Y \in C|X)$ on a fixed-length input. Consider a length-15 input X; denote its last 5 examples X'. We feed Llama X', but shift in-circuit heads' attention off the BOS token and onto content tokens. We then compute $(p_{int}(Y \in C|X') - p(Y \in C|X'))/(p(Y \in C|X) - p(Y \in C|X'))$, the normalized effect of this intervention. This intervention works with increasing efficacy as we intervene on more heads: intervening on the circuit recovers 71% of performance, while intervening on the top 10% or 15% of heads according to the circuit's ranking works better (Figure 6, right); see Appendix L for details. This mechanism is not unique to Number Game—it works in other in-context numeric reasoning tasks (see Appendix M).

Discussion. If Llama's growth in $p(Y \in C|X)$ is driven by attention that moves from the BOS token to other tokens, this suggests a functional form for its evidence accumulation. If for one head, the BOS token receives logit ℓ_B, and each new non-BOS token receives the same logit ℓ_O, the fraction of attention mass on non-BOS tokens after N tokens should grow hyperbolically as $A(N) = Ne^{\ell_O}/(Ne^{\ell_O} + e^{\ell_B}) = N/(N + e^{\ell_B - \ell_O})$. In reality, many heads engage in this behavior in tandem, and the growth in confidence is also mediated by neurons which can transform this signal further, so this is a simplification. Nonetheless, fitting the logits and activations of writer neurons to a hyperbolic form works notably better than using the earlier-proposed power-law form (Appendix L).

6 A New Model of Llama's Number Game Behavior

To conclude our investigation, we propose a Bayesian model that more closely reflects Llama's inference mechanisms: **(i)** it is a product of Bayesian experts (Hinton, 1999), which log-linearly pools beliefs from independent hypothesis families f (Winkler, 1968; Bordley, 1982), several of which can be true at once; **(ii)** its confidence grows hyperbolically with the evidence seen; and **(iii)** it is tolerant of noise in X with respect to a hypothesis h.

Llama embeds input numbers onto belief manifolds, one per hypothesis family (Figure 3). Our model has five such families H_f, plus a prior family (Appendix N). Each maps a number x to a position $\phi_f(x)$ on its manifold. For most families, we take $\phi_f(x)$ as a one-hot code (this equals a Fourier embedding with equally weighted harmonics, in a rotated basis). For intervals, $\phi_f(x) = (1, x, x^2)$, the sufficient statistics of a Gaussian. Llama averages the embeddings into a *state* and scales it hyperbolically with the evidence (Figure 6): $s_f(N) = \frac{\tau}{N+\tau} \sum_{i=1}^{N} \phi_f(x_i)$ **(ii)**. Hypothesis writer MLPs then approximate each family's log-predictive distribution $\log p(y|f, X)$. This requires a posterior $p(h|X, f)$, with learned log-likelihood weights γ_h and log-priors β_h, and a predictive $p(y|h)$ that is uniform over the members of h, but leaks a fraction ϵ of its mass to $p_0(y) \propto y^{-\lambda}$:

$$\log \widetilde{p}(h \mid X, f) = \underbrace{\gamma_h \langle s_f, \phi_f(h) \rangle}_{\text{Likelihood}} + \underbrace{\beta_h}_{\text{Prior}} \qquad p(y \mid h) = (1-\epsilon)\frac{\mathbb{1}[y \in h]}{|h|} + \epsilon\, p_0(y) \tag{3}$$

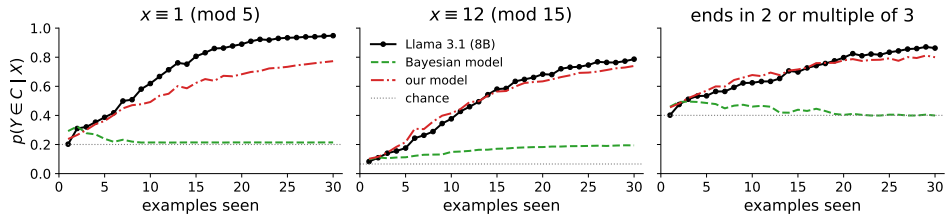


Figure 7: $p(Y \in C|X)$ predicted by our new model (red), the Bayesian model (green), and Llama (black), for 3 concepts C. Because our model includes $x = n \mod 5$ as a hypothesis, it can learn this (left). As it does not normalize over hypotheses, it can express hypotheses outside its hypothesis space via conjunction. It assigns non-zero probability to disjunctions, unlike the Bayesian model.

For one-hot families, $\langle s_f, \phi_f(h) \rangle$ is the fraction of X in h (for intervals, we use the Gaussian's marginal likelihood). A number outside h thus lowers h's score rather than eliminating it **(iii)**. Each family also has a background hypothesis $\varnothing$, scored by the log-likelihood of X under a uniform distribution. A hypothesis gets little weight until it outscores $\varnothing$; then its weight grows exponentially in the fraction of X in h, matching the threshold near 40% in Llama's logits (Figure 5, left). Each family marginalizes over its hypotheses, and so, as on Llama's manifolds, hypotheses compete within a family (*even* vs. *odd*) but combine into conjunctions across families (Section 3) **(i)**.

$$\log p(y|f, X) = \log \sum_{h \in H_f} p(y|h)\, p(h|X, f) \qquad \log p(y|X) = \sum_{f \in F} \log p(y|f, X) \tag{4}$$

With four more parameters than the Bayesian model (71 vs. 67), our model achieves lower KL (0.068 vs. 0.074). It also predicts Llama's behavior better on three kinds of new concepts: $x = 1 \mod 5$, and conjunctions and disjunctions of hypotheses from different manifolds. We expect the Bayesian model to fail on all three: it lacks $= n \mod 5$; its hypotheses compete in one posterior, so it cannot form conjunctions; and its indicator function eliminates both hypotheses in a disjunction. Indeed, our model beats it on all three (Figure 7). In sum, the new model is more complex than the Bayesian model, but it reflects the mechanisms underlying Llama's Number Game performance.

7 Discussion

Past work has stressed representational geometry as key to understanding and controlling LMs (Geiger et al., 2026); we show that Llama's belief geometry also yields quantitative predictions about its behavior. Llama's failure to output *multiples-of-7*, and ability to represent numbers equal to 12 (mod 15) stem directly from the limited set of features with which it represents its hypotheses, and the mechanism—conjunction of beliefs lying on orthogonal manifolds—it has to combine them.

This geometry likewise underlies Llama's uncertainty processing. Because Llama's beliefs lie on Fourier planes, it can represent ambiguity using averaging, a natural operation for transformers. Averaging two distinct features yields an intermediate belief between them, while averaging randomly distributed features yields a representation at the origin: no belief at all. In turn, studying ambiguous settings resolved a question about model geometry: why might Llama encode seemingly redundant features, like the third harmonic of a period-10 Fourier feature? This redundancy in fact allows Llama to represent an otherwise inexpressible ambiguity between *ends-in-4* and *ends-in-7*.

Llama updates its beliefs via a range of mechanisms. Some are task-specific, like context consistency circuits that check if an input is all even or odd. Others are more general: attention heads shift mass off of the BOS token as the input grows, increasing the effective number of observations.

To conclude, previous analyses of human behavior on the Number Game (Tenenbaum, 1999a) remained at Marr's computational level (Marr, 2010), as the underlying mechanisms were not accessible. In Llama, they are, allowing us to study the algorithmic and implementational levels directly. However, our insights were not confined to those levels; they informed a new computational model of Llama's behavior that outperformed the original Bayesian model and generalized to new settings. Bridging the gap between computational and implementational analyses can thus benefit both sides.

AI USAGE DISCLOSURE

The authors designed this paper's experiments and used AI to aid with implementing them. AI was used to find typos and provide general feedback on the main text, as well as to draft portions of appendices. All AI contributions have been edited to ensure factuality and human readability.

REFERENCES

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4895–4901, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.298. URL https://aclanthology.org/2023.emnlp-main.298/.

Aryaman Arora, Kirill Acharya, Nathan Hu, Yanzhe Zhang, Noah Goodman, Dan Jurafsky, and Christopher Potts. Matryoshka attribution: Learning to attribute language model outputs to representations and weights, 2026. URL https://arxiv.org/abs/2609.25518.

Arghavan Bazigaran and Hansem Sohn. Concept generalization in humans and large language models: Insights from the number game, 2025. URL https://arxiv.org/abs/2512.20162.

Nora Belrose, David Schneider-Joseph, Shauli Ravfogel, Ryan Cotterell, Edward Raff, and Stella Biderman. Leace: Perfect linear concept erasure in closed form. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 66044–66063. Curran Associates, Inc., 2023. doi: 10.52202/075280-2884. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/d066d21c619d0a78c5b557fa3291a8f4-Paper-Conference.pdf.

Eric Bigelow, Daniel Wurgaft, YingQiao Wang, Noah Goodman, Tomer Ullman, Hidenori Tanaka, and Ekdeep Singh Lubana. Belief dynamics reveal the dual nature of in-context learning and activation steering, 2026. URL https://arxiv.org/abs/2511.00617.

Eric J. Bigelow and Steven T. Piantadosi. Inferring priors in compositional cognitive models. In *Proceedings of the Cognitive Science Society*, 2016. URL http://colala.berkeley.edu/papers/bigelow2016inferring.pdf.

Robert F. Bordley. A multiplicative formula for aggregating probability assessments. *Management Science*, 28(10):1137–1148, 1982. ISSN 00251909, 15265501. URL http://www.jstor.org/stable/2630942.

Lawrence Chan, Adrià Garriga-Alonso, Nicholas Goldwosky-Dill, Ryan Greenblatt, Jenny Nitishinskaya, Ansh Radhakrishnan, Buck Shlegeris, and Nate Thomas. Causal scrubbing, a method for rigorously testing interpretability hypotheses. *AI Alignment Forum*, 2022. https://www.alignmentforum.org/posts/JvZhhzycHu2Yd57RN/causal-scrubbing-a-method-for-rigorously-testing.

Xander Davies, Max Nadeau, Nikhil Prakash, Tamar Rott Shaham, and David Bau. Discovering variable binding circuitry with desiderata, 2023. URL https://arxiv.org/abs/2307.03637.

Joshua Engels, Eric J Michaud, Isaac Liao, Wes Gurnee, and Max Tegmark. Not all language model features are one-dimensionally linear. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=d63a4AM4hb.

Javier Ferrando, Oscar Balcells Obeso, Senthooran Rajamanoharan, and Neel Nanda. Do i know this entity? knowledge awareness and hallucinations in language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=WCRQFlji2q.

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.

Atticus Geiger, Zhengxuan Wu, Christopher Potts, Thomas Icard, and Noah Goodman. Finding alignments between interpretable causal variables and distributed neural representations. In Francesco Locatello and Vanessa Didelez (eds.), *Proceedings of the Third Conference on Causal Learning and Reasoning*, volume 236 of *Proceedings of Machine Learning Research*, pp. 160–187. PMLR, 01–03 Apr 2024. URL https://proceedings.mlr.press/v236/geiger24a.html.

Atticus Geiger, Ekdeep Singh Lubana, Thomas Fel, Jack Merullo, Michael Byun, Owen Lewis, and Thomas McGrath. The world inside neural networks. *Goodfire Research*, May 2026. https://www.goodfire.com/research/the-world-inside-neural-networks.

Noah D. Goodman, Joshua B. Tenenbaum, Jacob Feldman, and Thomas L. Griffiths. A rational analysis of rule-based concept learning. *Cognitive Science*, 32(1):108–154, 2008. doi: https://doi.org/10.1080/03640210701802071. URL https://onlinelibrary.wiley.com/doi/abs/10.1080/03640210701802071.

Rhys Gould, Euan Ong, George Ogden, and Arthur Conmy. Successor heads: Recurring, interpretable attention heads in the wild. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=kvcbV8KQsi.

Thomas L. Griffiths, Nick Chater, Charles Kemp, Amy Perfors, and Joshua B. Tenenbaum. Probabilistic models of cognition: exploring representations and inductive biases. *Trends in Cognitive Sciences*, 14(8):357–364, 2010. ISSN 1364-6613. doi: https://doi.org/10.1016/j.tics.2010.05.004. URL https://www.sciencedirect.com/science/article/pii/S1364661310001129.

Thomas L. Griffiths, Jian-Qiao Zhu, Erin Grant, and R. Thomas McCoy. Bayes in the age of intelligent machines. *Current Directions in Psychological Science*, 33(5):283–291, 2024. doi: 10.1177/09637214241262329. URL https://doi.org/10.1177/09637214241262329.

Ritwik Gupta, Rodolfo Corona, Jiaxin Ge, Eric Wang, Dan Klein, Trevor Darrell, and David M. Chan. Enough coin flips can make LLMs act Bayesian. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7634–7655, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.377. URL https://aclanthology.org/2025.acl-long.377/.

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.

Jiatong Han, Jannik Kossen, Muhammed Razzak, and Yarin Gal. Semantic entropy neurons: Encoding semantic uncertainty in the latent space of LLMs. In *MINT: Foundation Model Interventions*, 2024. URL https://openreview.net/forum?id=BbZKxrZCNn.

Michael Hanna and Aaron Mueller. Incremental sentence processing mechanisms in autoregressive transformer language models. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3181–3203, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.164. URL https://aclanthology.org/2025.naacl-long.164/.

Michael Hanna, Ollie Liu, and Alexandre Variengien. How does GPT-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=p4PckNQR8k.

G.E. Hinton. *Products of experts*, pp. 1–6. IEE, 1999. doi: 10.1049/cp:19991075. URL https://digital-library.theiet.org/doi/abs/10.1049/cp%3A19991075.

Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:498–520, 1933. URL https://api.semanticscholar.org/CorpusID:144828484.

Jennifer Hu and Michael Frank. Auxiliary task demands mask the capabilities of smaller language models. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=U5BUzSn4tD.

Subhash Kantamneni and Max Tegmark. Language models use trigonometry to do addition. In *Workshop on Reasoning and Planning for Large Language Models*, 2025. URL https://openreview.net/forum?id=HH1hX4S0t8.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

Alexander Ku, Declan Campbell, Xuechunzi Bai, Jiayi Geng, Ryan Liu, Raja Marjieh, R. Thomas McCoy, Andrew Nam, Ilia Sucholutsky, Liyi Zhang, Jian-Qiao Zhu, and Thomas Griffiths. Levels of analysis for large language models. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 384(2320):20250012, 05 2026. ISSN 1364-503X. doi: 10.1098/rsta.2025.0012. URL https://doi.org/10.1098/rsta.2025.0012.

Dharshan Kumaran, Arthur Conmy, Federico Barbero, Simon Osindero, Viorica Patraucean, and Petar Veličković. How do LLMs compute verbal confidence? In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=t7oQT3uX66.

Amit Arnold Levy and Mor Geva. Language models encode numbers using digit representations in base 10. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pp. 385–395, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-190-2. doi: 10.18653/v1/2025.naacl-short.33. URL https://aclanthology.org/2025.naacl-short.33/.

Jack Lindsey, Wes Gurnee, Emmanuel Ameisen, Brian Chen, Adam Pearce, Nicholas L. Turner, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. On the biology of a large language model. *Transformer Circuits Thread*, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/biology.html.

Dong C. Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45:503–528, 1989. URL https://api.semanticscholar.org/CorpusID:5681609.

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

David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. The MIT Press, 07 2010. ISBN 9780262514620. doi: 10.7551/mitpress/9780262514620.001.0001. URL https://doi.org/10.7551/mitpress/9780262514620.001.0001.

Alexander Modell, Patrick Rubin-Delanchy, and Nick Whiteley. The origins of representation manifolds in large language models, 2025. URL https://arxiv.org/abs/2505.18235.

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*, 2023. URL https://openreview.net/forum?id=9XFSbDPmdW.

Yaniv Nikankin, Anja Reusch, Aaron Mueller, and Yonatan Belinkov. Arithmetic without algorithms: Language models solve math with a bag of heuristics. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=O9YTt26r2P.

nostalgebraist. interpreting gpt: the logit lens. LessWrong, 2020. URL https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens.

Maya Okawa, Bo Zhao, Eric Bigelow, Rose Yu, Tomer Ullman, Ekdeep Singh Lubana, and Hidenori Tanaka. Emergence of hierarchical emotion organization in large language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=vC8wORdy2x.

Sriram Padmanabhan, Kanishka Misra, Kyle Mahowald, and Eunsol Choi. On language models' sensitivity to suspicious coincidences, 2025. URL https://arxiv.org/abs/2504.09387.

Core Francisco Park, Andrew Lee, Ekdeep Singh Lubana, Yongyi Yang, Maya Okawa, Kento Nishi, Martin Wattenberg, and Hidenori Tanaka. ICLR: In-context learning of representations. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=pXlmOmlHJZ.

Mateusz Piotrowski, Paul M. Riechers, Daniel Filan, and Adam Shai. Constrained belief updates explain geometric structures in transformer representations. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=f6Hl60FBFU.

Kyle J. Ray, Paul M. Riechers, and Adam S. Shai. The geometry of nonergodic composition. *Belief Updates (Simplex Blog)*, 2026. URL https://belief-updates.pub/nonergodic-geometry/.

Raphaël Sarfati, Eric Bigelow, Daniel Wurgaft, Siddharth Boppana, 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.

Adam Shai, Loren Amdahl-Culleton, Casper L. Christensen, Henry R Bigelow, Fernando Rosas, Alexander Boyd, Eric A. Alt, Kyle J Ray, and Paul M. Riechers. Transformers learn factored representations. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=flDGB9On0h.

Adam S. Shai, Sarah E. Marzen, Lucas Teixeira, Alexander Gietelink Oldenziel, and Paul M. Riechers. Transformers represent belief state geometry in their residual stream, 2025. URL https://arxiv.org/abs/2405.15943.

Michal Štefánik, Timothee Mickus, Marek Kadlčík, Bertram Højer, Michal Spiegel, Raúl Vázquez, Aman Sinha, Josef Kuchař, Philipp Mondorf, and Pontus Stenetorp. Language models learn universal representations of numbers and here's why you should care. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 30663–30681, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1415. URL https://aclanthology.org/2026.acl-long.1415/.

Alessandro Stolfo, Ben Peng Wu, Wes Gurnee, Yonatan Belinkov, Xingyi Song, Mrinmaya Sachan, and Neel Nanda. Confidence regulation neurons in language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=0og7nmvDbe.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomput.*, 568(C), February 2024. ISSN 0925-2312. doi: 10.1016/j.neucom.2023.127063. URL https://doi.org/10.1016/j.neucom.2023.127063.

Joshua B. Tenenbaum. Rules and similarity in concept learning. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*, NIPS'99, pp. 59–65, Cambridge, MA, USA, 1999a. MIT Press.

Joshua B. Tenenbaum. A bayesian framework for concept learning, 1999b. URL http://hdl.handle.net/1721.1/16714.

Joshua B. Tenenbaum and Thomas L. Griffiths. Generalization, similarity, and bayesian inference. *Behavioral and Brain Sciences*, 24(4):629–640, 2001. doi: 10.1017/s0140525x01000061.

Robert L. Winkler. The consensus of subjective probability distributions. *Manage. Sci.*, 15(2): B–61–B–75, October 1968. ISSN 0025-1909. doi: 10.1287/mnsc.15.2.B61. URL https://doi.org/10.1287/mnsc.15.2.B61.

Daniel Wurgaft, Ekdeep Singh Lubana, Core Francisco Park, Hidenori Tanaka, Gautam Reddy, and Noah Goodman. In-context learning strategies emerge rationally. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=bBUUOQI0N6.

Daniel Wurgaft, Can Rager, Matthew Kowal, Sheridan Feucht, Usha Bhalla, Tal Haklay, Eric Bigelow, Raphaël Sarfati, Jack Merullo, Noah Goodman, Thomas Fel, Atticus Geiger, and Ekdeep Singh Lubana. Manifold steering reveals the shared geometry of neural network representation and behavior. In *Third Conference on Language Modeling*, 2026. URL https://openreview.net/forum?id=pf3HKAGT2M.

Ziqian Zhong, Ziming Liu, Max Tegmark, and Jacob Andreas. The clock and the pizza: Two stories in mechanistic explanation of neural networks. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=S5wmbQc1We.

Tianyi Zhou, Deqing Fu, Vatsal Sharan, and Robin Jia. Pre-trained large language models use fourier features to compute addition. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=i4MutM2TZb.

A Related Work

Bayesian Modeling and LMs. Our study draws on two bodies of work combining Bayesian models and LMs. One line of work uses Bayesian models as normative standards to which LMs are compared (Gupta et al., 2025); past work has evaluated Llama's behavior on the Number Game in that context. Padmanabhan et al. (2025) find that Llama succeeds only when given a prompt with task-relevant information, while Bazigaran & Sohn (2025) find that Llama fails to play at all. However, both studies directly ask LMs if a query number is in the concept, rather than having them predict the next number; we found this leads to worse performance. Other work has advocated for using Bayesian modeling as a tool for predicting and understanding LMs' behavior, especially in the context of in-context learning (Griffiths et al., 2024; Ku et al., 2026; Bigelow et al., 2026).

Mechanisms Underlying Uncertainty. Past work on mechanistic interpretability has studied uncertainty from various angles. Recent work has found neurons that modulate LMs' uncertainty via their overall or semantic entropy (Stolfo et al., 2024; Han et al., 2024), or found mechanisms by which LMs explicitly verbalize their uncertainty (Kumaran et al., 2026; Ferrando et al., 2025; Lindsey et al., 2025). Our study focuses on how LMs perform inference in situations where ambiguous evidence leads to uncertain beliefs. Past work has characterized and predicted the geometry of and mechanisms behind such beliefs in toy models (Shai et al., 2025; Piotrowski et al., 2025; Shai et al., 2026; Ray et al., 2026). Past work has also observed such beliefs in larger LMs, in the context of Gaussian distributions (Sarfati et al., 2026), and syntactic ambiguity (Hanna & Mueller, 2025).

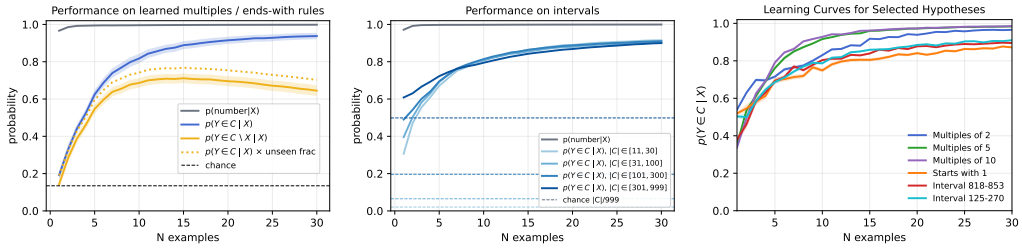


Figure 8: **Left**: Llama's behavior on inputs drawn from a rule that it successfully learns. It assigns increasing probability to numbers in C, and the probability it assigns to unseen numbers in C tracks the probability of a number in C times the fraction of numbers that remain unseen. **Center**: Learning curves for various intervals. Llama successfully learns interval hypotheses of varying widths, to similar levels of $p(Y \in C|X)$. **Right**: Llama learns different rules at different rates.

LM Representational Geometry. Recent work has uncovered complex structures within LMs' representations that they use to support their computations (Gould et al., 2024; Zhou et al., 2024; Park et al., 2025; Engels et al., 2025; Gurnee et al., 2025; Kantamneni & Tegmark, 2025; Levy & Geva, 2025; Modell et al., 2025; Okawa et al., 2026). Crucially, these structures are also causally relevant to model behavior, and can be steered upon in order to reliably manipulate model outputs (Sarfati et al., 2026; Feucht et al., 2026; Wurgaft et al., 2026).

B Full Behavioral Experiments

In this section, we provide more details regarding the behavioral Number Game experiments that we performed. First, note that unlike prior work, we consider all numbers 1-999 as potential targets in the Number Game. This provides LMs longer inputs from hypotheses with fewer members, like *multiples of 15*, without showing the LM all of the hypothesis' members, which would remove the need for generalizing to unseen members. This is important, as LMs generalize slowly and often need to see many examples.

Also note that we have the LM predict the next token, rather than answer a *yes/no* question about whether a particular number is likely to be generated. This ensures that task demands, like understanding instructions or formatting *yes/no* output, do not overshadow the model's basic task abilities (Hu & Frank, 2024). This is a small adaptation; Equation (1)'s model can be adapted to our setting simply by replacing $p(y \in C|h)$ with $p(y|h) = \mathbb{1}[y \in h]/|h|$.

Second, our dataset consists of inputs sampled from the following concepts: *multiples of n*, for $2 \leq n \leq 20$, as well as odd numbers; *ends-in-n*, for $1 \leq n \leq 10$ (note that *ends-in-0* is covered by multiples of 10); *starts-with-n* for $1 \leq n \leq 9$.

It also includes intervals; however, because the space of all intervals is large, we randomly sample 400 $[m, n], m \leq n$, rather than attempting to evaluate all intervals. We obtain the intervals by first sampling a width w from $[10, 700]$, and then sampling a valid center from $[1, 999 - w + 1]$.

For each (C, N) pair, we sample either all unique length-N inputs from C or 100 such inputs, whichever is smaller. Across all of these inputs, we compute the following metrics, which track the probability assigned to numbers, numbers in C, and unseen numbers in C respectively:

$$p(Y \in [1,999]|X) = \sum_{y \in 1 \ldots 999} p(y \mid X), \qquad p(Y \in C|X) = \sum_{y \in C} p(y \mid X), \text{ and}$$

$$p(Y \in C \setminus X|X) = \sum_{y \in C \setminus X} p(y \mid X).$$

We find that across concepts and N, $p(Y \in [1,999]|X)$ is high: Llama puts high probability on numbers from the very start, (Figure 8, left, center). For concepts that Llama learns successfully, $p(Y \in C|X)$ grows rapidly with the first few examples and continues to grow as Llama sees more examples (Figure 8, left). Moreover, this is not purely due to copying: $p(Y \in C \setminus X|X)$ also grows

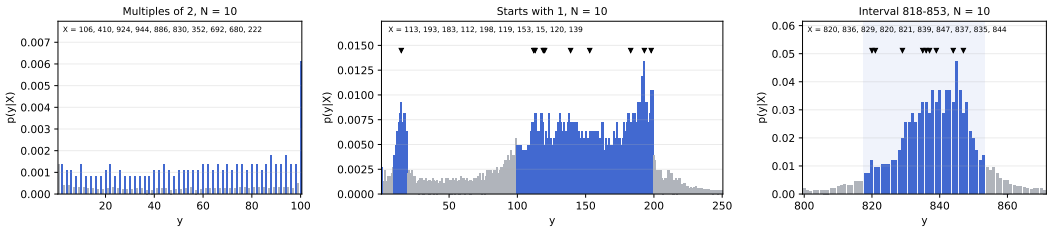


Figure 9: Next (number) token distributions from Llama given 10 inputs from a given concept C; numbers in C are colored blue. **Left**: Llama's behavior on *multiples of 2*. It assigns mostly uniform probability to all multiples of 2, but assigns high probability to 100, and some probability to 1, even though it is not a multiple of 2. **Center**: Despite only seeing one example in [10,19], Llama successfully upweights all numbers that lie between [10,19] and [100, 199], though it does not do so evenly, and does not appear to upweight 1. **Right**: Llama places most of its probability in the desired interval, [818,853], but the output distribution's shape is not a flat, uniform interval.

with N, and closely tracks $p(Y \in C|X) \cdot \frac{|C \setminus X|}{|C|}$, where the latter is the fraction of items in the ground truth concept C that remain unseen. Finally, Llama is able to predict interval concepts well (Figure 8, center).

If we look at Llama's behavior on individual rules, we can see that not all rules are learned identically. For example, *multiples-of-5/10* arrive at a much higher $p(Y \in C|X)$ than the interval rules, which also start with a lower $p(Y \in C|X)$ (Figure 8, right). *Starts-with-1* has a higher $p(Y \in C|X)$ initially, but the lowest at the very end.

Finally, if we look at Llama's behavior on individual inputs, we see just how noisy it is (Figure 9). While it succeeds in aggregate at performing the Number Game, and does not only use copying to do so, the precise distribution of its probability mass across numbers in a given ground-truth concept C is highly irregular, and unlikely to be predicted by any standard model. Indeed, its interval outputs look strikingly like a continuous distribution, and its *starts-with-n* outputs do too. This is consistent with the interval representations matching those of Gaussians from Sarfati et al. (2026).

B.1 Other Hypotheses

Llama can represent other hypotheses not discussed in the main text. Hypotheses such as *prime numbers*, *square numbers*, and *powers of 2* are all common hypotheses in the Number Game literature, and Llama can learn them. We omit them primarily because their representational geometry is not very interesting: preliminary investigations suggested that these were ad-hoc directions representing memorized sets of numbers.

C Full Bayesian Experiments

We start our full explanation of the Bayesian model by re-deriving the model. For an input X and next token y, the Bayesian observer with hypothesis space $\mathcal{H}$ computes:

$$p(y|X) = \sum_{h \in \mathcal{H}} p(y|h)p(h|X) \propto \sum_{h \in \mathcal{H}} \frac{\mathbb{1}[y \in h]}{|h|} p(X|h)p(h) = \sum_{h \in \mathcal{H}} \frac{\mathbb{1}[y \in h]}{|h|} \left(\frac{\mathbb{1}[x \in h]}{|h|} \right)^{|X|} p(h), \tag{5}$$

assuming that each x is sampled from the hypothesis space uniformly at random. Our model is nearly identical, learning log-priors and log-likelihoods, and using $|X|^{1-\alpha}$ instead of $|X|$ to account for the slower, power-law evidence accumulation observed by Wurgaft et al. (2025) and Bigelow et al. (2026):

$$p_B(y|X) \propto \sum_{h \in \mathcal{H}, y \in h} \frac{\mathbb{1}[X \subseteq h]}{|h|} \exp(\gamma_h |X|^{1-\alpha} + \beta_h) \tag{6}$$

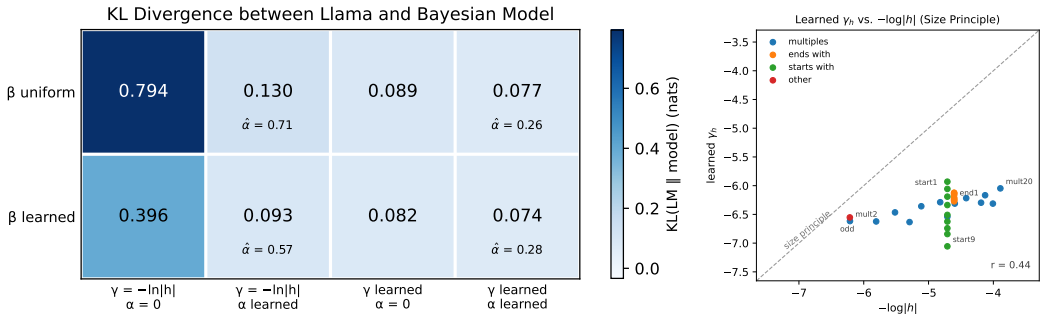


Figure 10: **Left**: Quality of Bayesian model fits using a population loss, with each parameter free or pinned to its default. **Right**: γ_h vs. $-\log|h|$ for various rules in the α, β, γ-learned setting indicates that the two often differ and are only weakly correlated.

We set our hypothesis space $\mathcal{H}$ to be identical to the ground-truth concept space, omitting unlearned *multiples-of-n* rules. We also add a hypothesis $\tilde{h}$ to model Llama's small-number bias and ensure full support over 1-999; we set $p_B(y|\tilde{h}) \propto y^{-p_q}$, with a learned parameter p_q. Unlike the other hypotheses, $\tilde{h}$ is consistent with every list. Its log-posterior is $\log p_B(\tilde{h} \mid X) = \beta_{\tilde{h}} + |X|^{1-\alpha} \cdot \frac{1}{|X|} \sum_{x \in X} \log p_B(x \mid \tilde{h}) - \log Z(X)$, so it gains weight when X contains small numbers.

We would like to learn parameters per-hypothesis, but the intervals cause problems; with nearly 500,000 interval hypotheses, it is unwieldy to learn per-hypothesis parameters. Instead, we follow Bigelow & Piantadosi (2016), using a hierarchical model where we first sample the type of hypothesis—interval or not—based on a prior $p(\text{interval})$, for which we learn a parameter π. All intervals receive a uniform prior, and their γ_h is taken to be $-\log|h|$, by the size principle.

For each of the non-interval hypotheses, we can learn β_h, γ_h per h, and compare to default Bayesian settings: β_h should be uniform, $\gamma_h = -\log|h|$, matching the size principle, and $\alpha = 0$ for linear evidence accumulation. We can also fix each of these to its Bayesian default, and see how this affects our model's performance.

Let p denote our LM and p_B the Bayesian model. As Llama's per-example outputs are noisy, we model its population-level outputs, i.e. its average output given N examples of a concept C:

$$\mathcal{L} = \sum_{C \in \mathcal{C}, 1 \leq N \leq 30} \text{KL}\left(\mathbb{E}_{X \sim (C,N)} p(y|X) || \mathbb{E}_{X \sim (C,N)} p_B(y|X)\right) \tag{7}$$

We fit the Bayesian model with Adam (Kingma & Ba, 2017).

For the data, we draw on the same dataset that we evaluated the model with originally. The concept space $\mathcal{C}$ consists of the 33 hypotheses from *multiples-of-n*, odd, *ends-in-n*, and *starts-with-n*. We divide the 400 intervals from the previous section into 6 bins based on their width: the width bins are 21–50, 51–100, 101–200, 201–300, 301–500 and 501–700. For each of these concepts C and for each $N = 1, \ldots, 30$, we create 100 length-N training lists and 100 held-out lists. We draw N numbers uniformly with replacement from C. If C is an interval bin, we first fix an interval that falls in that width bin before drawing numbers from it.

Our results (Figure 10, left) show that we achieve low KL divergence. The Bayesian fit also yields insights into model behavior. It suggests that Llama accumulates evidence sub-linearly ($\hat{\alpha} = 0.28$), and fails to follow the size principle exactly, as $\gamma_h = -\log|h|$ never holds; see the divergence of the points from the line in Figure 10, right.

We can even investigate the learned log-likelihoods γ_h, in order to find concrete failures of the size principle. Taking the γ_h values from the condition where all parameters are learned, we see that $\gamma_{mult-9}(-6.54) < \gamma_{mult-6}(-6.36)$, predicting that Llama will assign more probability to multiples of 6 than to multiples of 9 when presented with evidence compatible with both, contrary

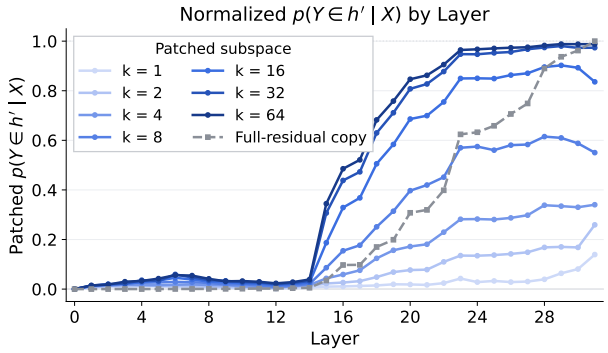


Figure 11: Normalized $p(Y \in h'|X)$ when using learned DAS subspaces to patch a hypothesis h' onto an input X drawn from another hypothesis h; we also include full residual patching as a baseline. DAS subspaces only start to become effective at layer 15, and are fully effective at 32D.

to the size principle. Empirically, this is true: given 5 multiples of 18, Llama assigns the most probability to multiples of 18 (0.38), but more probability to multiples of 6 (0.27) than 9 (0.08).

D Hypothesis Space Geometry Details

D.1 Distributed Alignment Search

Distributed alignment search (DAS; Geiger et al., 2024) is a technique that learns the optimal subspace for performing a given patching operation. Concretely, let X, X' be inputs to a model p_θ, which produce hidden activations $\mathbf{z}, \mathbf{z}'$ at a given place in the model; furthermore, let R be a rank-k projector that acts as the orthonormal basis the k-dimensional subspace $RR^\top$, and $\mathcal{L}$ be a loss function. Denote by $p_{\mathbf{z}\leftarrow\mathbf{z}'}(Y|X)$ the next-token distribution when we patch the subspace defined by R of $\mathbf{z}'$ onto that subspace of $\mathbf{z}$, i.e., $p_\theta(X \mid \text{do}(\mathbf{z} \leftarrow \mathbf{z} + \mathbf{R}\mathbf{R}^\top(\mathbf{z}' - \mathbf{z})))$. Then DAS learns R to minimize $\mathcal{L}(X, X', p_{\mathbf{z}\leftarrow\mathbf{z}'}(Y|X))$, i.e. to minimize some loss function that tracks the model's behavior after this patch. Typically, the loss function measures the similarity between the model's behavior upon performing this patch, and its behavior on X', but DAS is flexible, and this need not be the case.

In our particular case, X, X' are inputs drawn from distinct hypotheses h, h', and our loss function is the KL divergence between Llama's output on X when we perform patching with R, and its output on X'. Regarding the location of the activation $\mathbf{z}$, we sweep over the (last-position) residual stream at every layer; we distinguish between the residual stream after each layer's attention and after its MLP. We sweep $k = 1, 2, 4, 8, 16, 32, 64$, and train for 3000 steps with a batch size of 24 (X, X') pairs. R is initialized with top-k right singular vectors of the pooled $\mathbf{z} - \mathbf{z}'$ differences. We keep R orthonormal using QR factorization after each Adam step. We use a learning rate of 0.01.

For our dataset, we use length-10 inputs drawn from each of the hypotheses that Llama learns. That is, our data consists of inputs drawn from the learned *multiples-of-n* hypotheses (and *odd*), all *ends-in-n* hypotheses, 400 interval hypotheses, and 9 *starts-with-n* hypotheses. When training the DAS subspace, we sample pairs h, h' independently (we resample if $h = h'$), but we weight the probability of seeing each group: 40% for *multiples-of-n*, *ends-in-n*, and *odd* hypotheses; 40% for interval hypotheses, and 20% *starts-with-n* hypotheses. This ensures that all of the groups of hypotheses that we care about are well-captured by the DAS subspace. Additionally, within interval hypotheses, we first sample a width from the bins [21-50, 51-100, 101-200, 201-300, 301-500, 501-700], and then sample an interval whose width falls in that bin. This ensures width diversity within the intervals.

After training, we evaluate the DAS subspaces. Given some X, X' from hypotheses h, h', our metric is $\frac{p_{\mathbf{z}\leftarrow\mathbf{z}'}(Y\in h'|X) - p(Y\in h'|X)}{p(Y\in h'|X') - p(Y\in h'|X)}$, a version of $p(Y \in h'|X)$ that is normalized to lie between 0 and 1, indicating whether Llama's patched behavior matches its behavior on X or X'. Our results

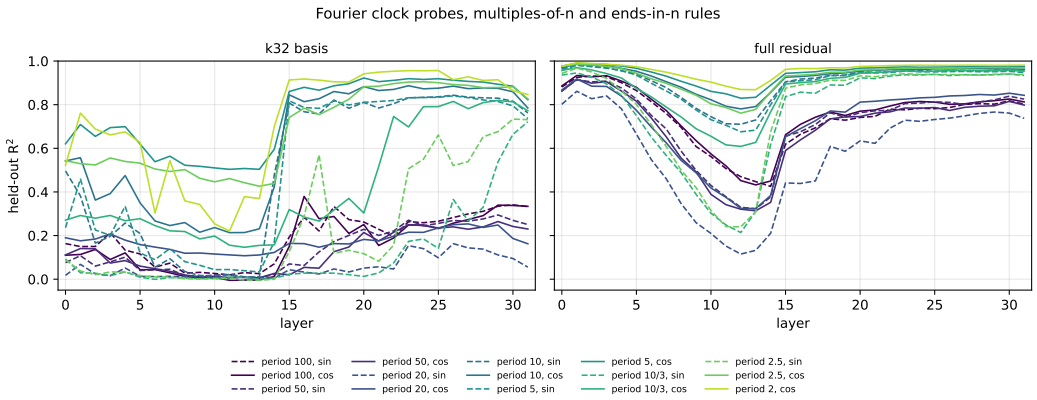


Figure 12: Probing results for short-period / high-frequency Fourier features, as found in *multiples-of-n* and *ends-in-n*. These are trained on representations from the 32D DAS subspace (left) and full residual stream (right).

(Figure 11) indicate that the emergence of patchable beliefs is a gradual process. Patching begins to be effective starting at layer 15, but is not fully effective until layer 23. A subspace of rank $k = 32$ suffices to recover as much as the next largest k, though smaller subspaces offer partial recovery. DAS outperforms a full-residual patch at every layer, suggesting that DAS finds a subspace that excludes the noise or contradictory signals that a full residual patch might have introduced.

D.2 *Multiples-of-n* AND *Ends-with-n*

To find the features representing *multiples-of-n* and *ends-with-n* hypotheses, we use Fourier probes (Levy & Geva, 2025; Štefánik et al., 2026; Feucht et al., 2026). Such probes leverage the fact that LMs (and other neural models) often encode numbers via circular representations that ultimately reflect periodic functions of their values (Hanna et al., 2023; Nanda et al., 2023; Zhong et al., 2023; Gould et al., 2024; Kantamneni & Tegmark, 2025). For example, Feucht et al. use Fourier probes to show that, given inputs like "$a + b =$", Llama-3.1 (8B) represents the value of $a + b$ using Fourier features at the final position of its input. Given a target number x and its representation $\mathbf{z}^\ell$ at layer ℓ in the model, a Fourier probe typically learns weights $\mathbf{w}, b$ to predict

$$\mathbf{w}_{sin}^{\ell,T}\mathbf{z}^\ell + b_{sin}^{\ell,T} = \sin\left(\frac{2f\pi x}{T}\right); \quad \mathbf{w}_{cos}^{\ell,T}\mathbf{z}^\ell + b_{cos}^{\ell,T} = \cos\left(\frac{2f\pi x}{T}\right),$$

where T is the period and f the frequency of the representation. In contrast with prior work, we do not probe directly for the value of any individual x; rather, because we believe beliefs to be constructed as the average of per-input features, we instead probe for the mean value of the Fourier features of the input numbers:

$$\mathbf{w}_{sin}^{\ell,T}\mathbf{z}^\ell + b_{sin}^{\ell,T} = \frac{1}{|X|}\sum_{x\in X}\sin\left(\frac{2f\pi x}{T}\right); \quad \mathbf{w}_{cos}^{\ell,T}\mathbf{z}^\ell + b_{cos}^{\ell,T} = \frac{1}{|X|}\sum_{x\in X}\cos\left(\frac{2f\pi x}{T}\right)$$

.

We train these probes on length-10 inputs drawn from *multiples-of-n* and *ends-with-n* hypotheses, on each of the space positions starting at the 3rd (to avoid training at locations where the model may yet be unsure of the task). We train inside our rank-32 DAS subspace, but compare to full-residual training as well; the latter matches the setup of prior work like Feucht et al. (2026), but later experiments show only the former probes to be causally effective. We use $T = 2, 5, 10, 20, 50, 100$, and $f = 1$. We also probe for $T = 5$ and $f = 2$ (equivalently $T = 2.5$), as well as $T = 10$ and $f = 3$ (equivalently $T = 3.33$), based on findings that Llama also uses higher-order harmonics of periods 5 and 10 to represent these hypotheses (Appendix K).

Our results (Figure 12) indicate that Llama indeed uses $T = 2, 5, 10$ for these hypotheses but not $20, 50, 100$. This makes sense: these rules are all unspecified with respect to these longer periods.

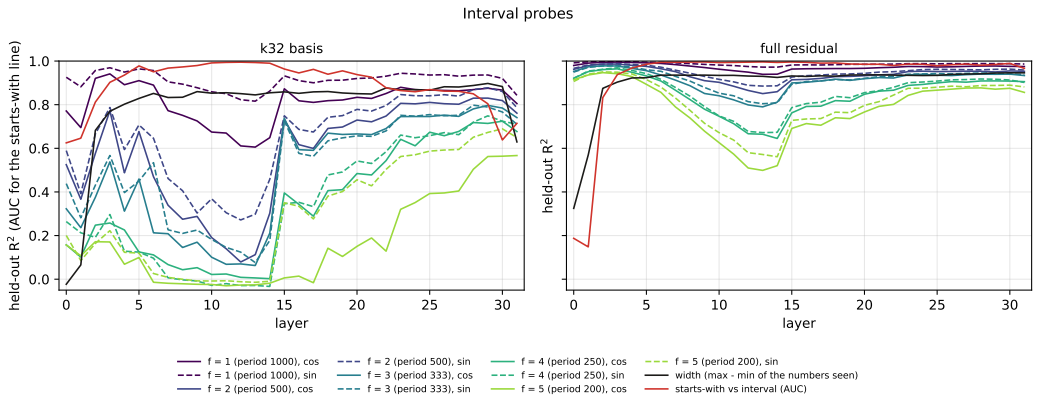


Figure 13: Probing results for long-period / low-frequency Fourier features, as found in intervals. These are trained on representations from the 32D DAS subspace (left) and full residual stream (right).

The $f \neq 1$ probes struggle more than the other low-period probes, with a gap between the sine and cosine probes' performance, though they notably outperform the higher-period probes. More generally, R^2 is low for early-mid layers when using the DAS subspace, but high and smoothly falling, then rising toward layer 15 in the full residual stream.

D.3 Intervals and *Starts-with-n*

To find Fourier features that encode intervals, we sample length-10 probing contexts from a set of hypotheses consisting of the 400 intervals from the behavioral section. For the purpose of *starts-with-n* probing, we also include the nine width-100 blocks $100n$ to $100n + 99$, 40 contexts per set, and 60 contexts per digit n from the *starts-with-n* set, drawn so that at least one one-/two-digit member appears. We then probe for the mean Fourier feature of these inputs using periods $T = 1000, 500, 333, 250, 200$; equivalently, one can consider this to be $T = 1$ and $f = 1, 2, 3, 4, 5$.

Our results (Figure 13) show that R^2 is poor for early-mid layers when using the DAS subspace, but high and smoothly falling, then rising toward layer 15 in the full residual stream. Of note, however, is that not all periods are eventually well recovered. In particular, the shorter period (200) / higher frequency of the interval features are not well recovered in the DAS subspace, though they are in the full residual stream.

What should we make of these discrepancies? The high accuracies of full residual stream probes in the early layers seem attributable to unused but decodable information: causal interventions at these layers (at the space position) do nothing. The conclusions are less clear as concern features that are hard to decode within the DAS subspace at later layers. These features are either unimportant overall, or unimportant enough that the DAS subspace can enable patching while excluding them. In the following section, we will see that the DAS subspace actually contains a mixture of these features.

What of *starts-with-n*? At first glance, intervals and *starts-with-n* might appear to be very different. In Figure 14 we take class-conditional mean representations of intervals (where a class is defined by the interval's center and width) and *starts-with-n* (where the class is defined by n) and perform PCA on these (Hotelling, 1933). The former appear as a 2D manifold - either a sheet, or a cone, whereas the latter appears as a curve. But *starts-with-n* hypotheses are fundamentally very similar to intervals: they consist of a width-100 interval centered at $100n+50$, plus a smaller width-10 interval at $10n + 5$, along with n itself. When one draws from a *starts-with-n* hypothesis h, the most likely outcome is something from the largest interval, which contains 100/111 of h's members. Given even a length-10 X sampled from h, 35% of Xs will have only members in $[100n, 100n + 99]$.

That is to say that oftentimes, *starts-with-n* and *intervals* representations are literally indistinguishable. This enables us to model these hypotheses well using essentially the same machinery that we

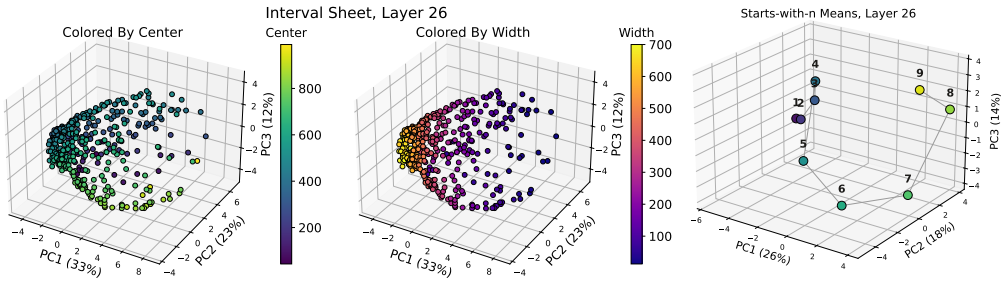


Figure 14: 3D PCA plots of representations of interval inputs (left) and *starts-with-n* inputs (right), within the DAS subspace.

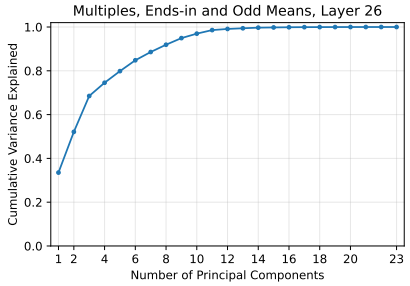


Figure 15: Cumulative variance explained vs. number of principal components for *multiples-of-n / ends-in-n* hypotheses

use for intervals in Appendix E.1; those results (Figure 18) show that there exists essentially one shared axis of difference between these types.

E Representation Steering and Interventions

All of the hypothesis representation manifolds that we find in Section 3 are causally involved in Llama's processing, and thus amenable to steering interventions. To do this, we first explicitly model the manifolds (Appendix E.1), allowing us to empirically predict where a hypothesis' representation should lie on its own manifold, and within the 32D DAS subspace. Then, we perform interventions, patching using both empirical activations and the ones predicted by our models (Appendix E.2).

E.1 Explicitly modeling manifolds

Before performing steering, we first build explicit models of the representational geometry that we found in each layer of the model.

Multiples-of-n / Ends-in-n To model the full structure of the *multiples-of-n/ends-in-n/odd* manifold, we first find its linear embedding dimension, i.e. the dimension of the smallest affine subspace that contains the manifold. Once we have found this, we can interpret the manifold in this subspace, and model this subspace alone.

To do so, we take 24 hypotheses: multiples of n for $n \in \{2, 3, 4, 5, 6, 8, 9, 10, 12, 15, 16, 18, 20\}$, numbers that end in $n = 0, \ldots, 9$, and odd numbers. For each hypothesis h, we construct 40 length-10 inputs by sampling 10 members from h. We then collect the layer 26 32-dimension DAS-subspace representations of these inputs (at the final space token) and average over the 40 examples, to create an averaged representation of h. We then perform PCA on these 24 representations and observe the explained variance as a function of the number of PCs.

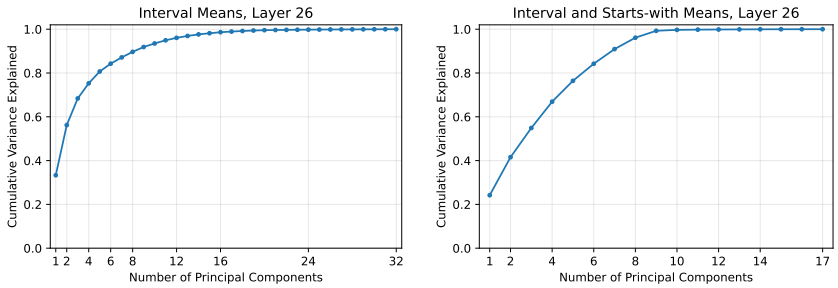


Figure 16: Cumulative variance explained vs. number of principal components for interval (left) and *starts-with-n* (right) hypothesis representations in the layer 26 32D DAS subspace. Though both hypotheses appear to be plausible manifolds in 3D, substantially more PCs are required to capture their full structure.

Our results (Figure 15) indicate that these hypotheses occupy an 11-dimensional subspace of the DAS subspace. We thus attempt to model them using the features that we identified earlier. Recall that in Appendix D, we probed for a variety of periods. The most effective of these had period $T = 2, 5/2, 10/3, 5, 10$; equivalently, these were period-10 probes with $f = 1, \ldots, 5$, i.e. the first 5 harmonics of a period-10 encoding. We also include features that identify specifically multiples of 3 and 9, and one that accounts for number of times that 2 appears as a factor of x, denoted $\nu_2(x)$. That is, our model for the representation of a hypothesis h in the DAS subspace is the following:

$$\hat{\mathbf{z}}(h) = \mathbf{b}_\emptyset + \sum_{f=1}^{5} \left[\mathbf{b}_{\cos,f} \frac{1}{|h|} \sum_{x \in h} \cos\big(2f\pi x/10\big) + \mathbf{b}_{\sin,f} \frac{1}{|h|} \sum_{x \in h} \sin\big(2f\pi x/10\big) \right] + \mathbf{b}_3 \frac{1}{|h|} \sum_{x \in h} \mathbb{1}[x \mod 3 = 0] + \mathbf{b}_9 \frac{1}{|h|} \sum_{x \in h} \mathbb{1}[x \mod 9 = 0] + \mathbf{b}_2 \frac{1}{|h|} \sum_{x \in h} \nu_2(x). \tag{8}$$

At layer 26, across the aforementioned dataset, our model achieves an R^2 of 0.92.

Intervals To model the full structure of the interval manifold, we first find its linear embedding dimension, just as in the prior section. Once we have found this, we can interpret the manifold in this subspace, and model this subspace alone.

To do so, we first sample 400 intervals: we first uniformly sample width (10-700) and then a valid center. We then construct 40 10-example inputs per interval. We next collect the layer 26 32-dimension DAS-subspace representations of these inputs (at the final space token), and average over the 40 examples, to create an averaged representation of that interval. We then perform PCA on these representations and observe the explained variance as a function of the number of PCs.

We find (Figure 16, left) that 6 PCs capture 84.3% of the variance, with further PCs yielding small improvements $\leq 2.5\%$, so we focus on interpreting those first. To do so, we visualize each PC with respect to the center of the interval (Figure 17), where each dot records an interval's center and value in the given principal component. This reveals clear periodic structure: every PC is periodic in the center of the interval in question.

The encoding of width is the most surprising aspect. In the first two PCs, we see that width is encoded explicitly: the value of an interval in these PCs is shifted downward as its width grows. However, in the higher PCs, the encoding changes: wider intervals correspond to waves whose amplitudes are attenuated or even reversed. This can be explained mechanistically and mathematically as a natural consequence of increasing width, rather than as explicit encoding of width.

We posit (with evidence in Appendix H) that Llama constructs the *center* component of an interval representation by using attention heads to take an average of the input numbers' representations. These numbers are also encoded with Fourier features, so we must take the average of said Fourier

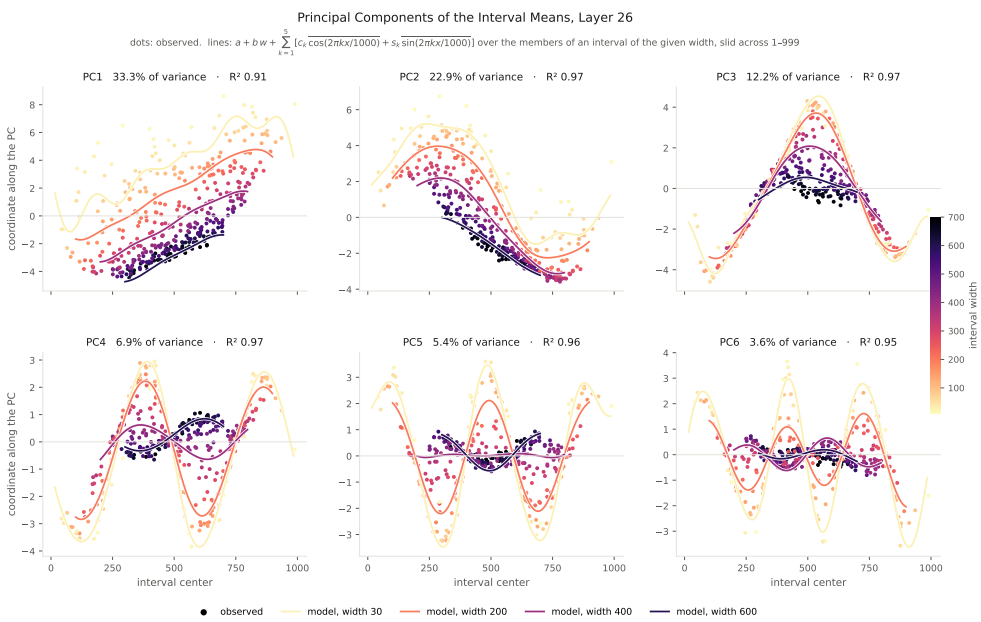


Figure 17: PC value vs. center for PCs 1-6 of the 32D DAS subspace at layer 26, for interval hypotheses. Each dot is a representation created by averaging 40 length-10 representations of a given interval. Each line corresponds to predictions made by our model of intervals at a given width. Our model captures all PCs well for intervals of various centers and widths.

features, rather than take the Fourier feature corresponding to the average or center of the input numbers. This leads to a mathematical explanation: as the width of an interval increases, we take averages over increasingly broad windows of numbers; doing so naturally causes the amplitude to shift with width. We note, however, that width is still encoded explicitly in the first two PCs, and is not just an artifact of this phenomenon.

To demonstrate that our explanation is correct, we construct an explicit model of these 6 PCs:

$$\hat{\mathbf{z}}(h) = \mathbf{b}_\emptyset + \mathbf{b}_w w + \sum_{f=1}^{5} \Big[\mathbf{b}_{\cos,f} \frac{1}{|h|} \sum_{x \in h} \cos\big(2f\pi x/1000\big) + \mathbf{b}_{\sin,f} \frac{1}{|h|} \sum_{x \in h} \sin\big(2f\pi x/1000\big) \Big] \tag{9}$$

where h is the interval hypothesis we are modeling, w is its width, and each $\mathbf{b} \in \mathbb{R}^6$ is a learned parameter vector. Across the same 400 averaged interval representations from before, we predict $\mathbf{z}$, the 6D subspace we identified in this section, and plot these predictions on Figure 17. We achieve an R^2 of 0.946 within the 6 PCs. Notably, dropping the explicit width term mainly hurts R^2 on the first two PCs, substantiating our claim that width is encoded only implicitly in the later PCs.

***Starts-with-n* Curves** We begin modeling the *starts-with-n* hypotheses in the same way as with the interval hypotheses: we compute the PCA of these hypotheses and look at explained variance. We construct two different kinds of length-10 *starts-with-n* hypothesis examples: those that contain at least one number $< n \times 100$, and those that do not. Note the latter are indistinguishable from width-100 interval hypotheses with center $n \times 100 + 50$, so including these in our analysis will allow us to find the dimensions by which Llama distinguishes between those intervals and *starts-with-n*. We sample 40 examples for each category, for each value of $n = 1, \ldots, 9$.

The explained variance analysis (Figure 16, right) shows that only 9 PCs are needed to capture all of the variance; this reflects the fact that we are considering the PCA of only 18 points, as opposed



Figure 18: PC value vs. center for PCs 1-9 of the 32D DAS subspace at layer 26, for *starts-with n*. Each dot is a representation created by averaging 40 length-10 representations of a *starts-with-n* input, either containing at least number $\leq 100 \times n$ (orange), or not (blue); the latter are indistinguishable from pure intervals. For intervals of various centers and widths model captures all PCs well but PC9; note that this is because each model is scored within-family, so the model receives no credit for capturing the interval-*starts-with-n* gap.

to the 400 intervals. Based on this, we choose to analyze all 9 PCs, on the basis that the cost of understanding all of the PCs is in this case not very high.

We perform the same analysis as before, plotting *starts-with-n* representations' values on each PC against n itself. We find (Figure 18) that these are once more very sinusoidal, this time in n. Notably, only in PC9 is there a large difference between *starts-with-n* with and without numbers less than 100; in the others, they are the same. To model the *starts-with-n* hypotheses, we can use the following model, which uses parameters adapted from the prior interval model:

$$\hat{\mathbf{z}}(h) = \mathbf{b}_{\emptyset} + \mathbf{b}_{w}\,|h| + \mathbf{b}_{\text{start}}\,\mathbb{1}[h \in \mathcal{H}_{\text{start}}] + \sum_{f=1}^{5}\Big[\mathbf{b}_{\cos,f}\,\frac{1}{|h|}\sum_{x\in h}\cos\big(2f\pi x/1000\big) + \mathbf{b}_{\sin,f}\,\frac{1}{|h|}\sum_{x\in h}\sin\big(2f\pi x/1000\big)\Big], \tag{10}$$

where $h_n = \{n\} \cup \{10n, \ldots, 10n{+}9\} \cup \{100n, \ldots, 100n{+}99\}$. Here, each parameter in the model except for b_{start} was taken directly from the prior model (Equation (9)) and expressed in the basis of

starts-with-n PCs. That is, this model is mostly zero-shot transfer from the intervals. The R^2 of this is 0.975 for pure intervals and 0.949 for *starts-with-n* including at least one out-of-interval number (where we learn an extra $\mathbf{b}_{start}$ constant offset).

E.2 INTERVENTIONS

Having confirmed where these manifolds do and do not exist, we can begin to intervene on them, which works as follows. Let R^ℓ be the rank-32 DAS subspace that we learned at layer ℓ, and let $\mathbf{z}^\ell$ be the residual stream at that layer, on an input X corresponding to hypothesis h, and let $\mathbf{c} = R^\top \mathbf{z}$. To intervene, we compute a new $\mathbf{z}_{int} = \mathbf{z} + \alpha R(\mathbf{c}' - \mathbf{c})$, where $\mathbf{c}'$ is the representation of the hypothesis h' we wish to steer toward, and α is a steering strength. We insert $\mathbf{z}_{int}$ in the place of $\mathbf{z}$.

Patching from examples We can perform patching interventions by setting $\mathbf{c}'$ to its value on a concrete other example X'. If we set $\alpha = 1$ and $\mathbf{c}' = R^\top \mathbf{z}'$, where $\mathbf{z}'$ is the residual stream representation of input X' drawn from hypothesis h', we recover the intervention on which DAS was trained. We can also constrain $\mathbf{c}'$ such that it differs from $\mathbf{c}$ only in the dimensions that we observed to be relevant for the given hypothesis. For example, if we want to steer from one Fourier-clock hypothesis h to another (h'), we can construct $\mathbf{c}'$ such that it only differs in the top 11 PCs of all of the Fourier-clock manifolds, rather than differing in all 32 dimensions. Formally, this is $\mathbf{c}' = \mathbf{c} + Q(R^\top \mathbf{z}' - \mathbf{c})$, where Q is a matrix selecting the relevant subspace.

Patching from a model of the *multiples-of-n* and *ends-in-n* manifold We directly compute $\mathbf{c}'$ as $\hat{\mathbf{z}}(h')$ according to Equation (8), the model developed in the previous section. The model is fit per coordinate of R, so $\hat{\mathbf{z}}(h')$ is a full 32-dimensional target; as above, Q can restrict the edit to the PCs analyzed in Appendix E.1.

Patching from a model of the interval manifold We directly compute $\mathbf{c}'$ as $\hat{\mathbf{z}}(h')$ according to Equation (9), the model of the interval manifold developed in the previous section. The model is fit per coordinate of R, so $\hat{\mathbf{z}}(h')$ is a full 32-dimensional target; as above, Q can restrict the edit to the PCs analyzed in Appendix E.1.

Patching from a model of the *starts-with-*n curve We directly compute $\mathbf{c}' = \hat{\mathbf{z}}(h')$ from Equation (10), which adds the learned offset $\mathbf{b}_{\text{start}}$ for starts-with-n. As above, the model is fit per coordinate of R, but Q can restrict the edit to the PCs analyzed previously.

Patching Experiment and Results Given this patching methodology, we perform the following experiment. As base inputs (onto which we will patch), we consider all learned *multiples-of-n* (2, 3, 4, 5, 6, 8, 9, 10, 12, 15, 16, 18, 20) and *ends-in-n* rules (the Fourier hypotheses), as well as all 9 *starts-with-n* rules. For intervals, we sample 400 intervals as in the prior section. For each of these we create 20 10-shot contexts. These same inputs / rules also act as targets to which we will push the base's output. We patch only within families that use the same manifolds, so we patch between *multiples-of-n* and *ends-in-n*, and from one interval to another, but not between the two. We consider all 720 Fourier pairs, 200 samples of interval pairs, and all 72 pairs of *starts-with-n* hypotheses, but filter out any pairs where the $p(Y \in C|X)$ for the steered-from and steered-to inputs is similar (within 0.2).

For each family, we consider 5 distinct interventions. The first two, we have seen before: full residual patching, DAS-subspace patching. The third is relevant DAS-subspace patching, e.g. only patching the Fourier planes if we are patching Fourier-family hypotheses. Finally, we perform patching, but construct the patched version analytically, using the previously discussed models for computing $\mathbf{c}'$, rather than patching. Our models can be fit to predict all of the 32D DAS subspace, or just the one that is relevant for the subspace; we report results for both setups. For these experiments, we report results at $\alpha = 1$ and $\alpha = 2$. We measure performance via recovery towards target, i.e. $(p_{h'}(X|\text{do(steer)}) - p_{h'}(X))/(p_{h'}(X') - p_{h'}(X))$, where $p_{h'}(X)$ is the probability assigned to members of the steered-towards hypothesis h' given the base input X from hypothesis h, X' is the corresponding input from h', and $p_{h'}(X|\text{do(steer)})$ is the probability assigned to members of h' under this steering intervention.

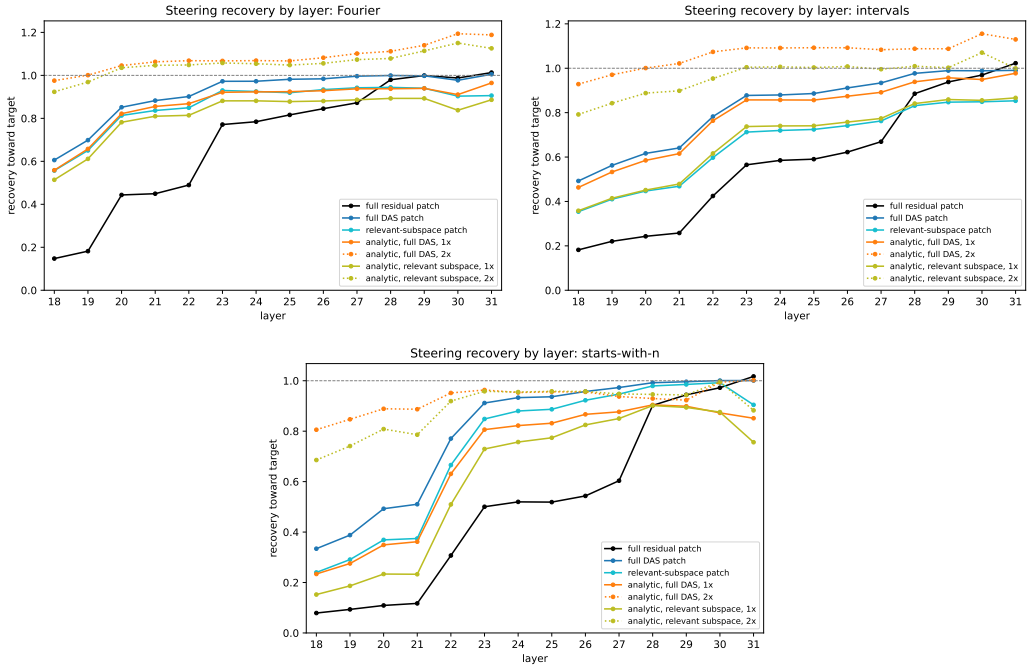


Figure 19: **Left**: Steering results for *multiples-of-n* and *ends-in-n*. **Right**: Steering results for intervals. **Bottom**: Steering results for *starts-with-n*

Our results indicate that steering is broadly successful. As seen previously, DAS subspace steering outperforms residual stream patching at most (early) layers. For the short-period Fourier features and the intervals, the relevant-subspace patch is seldom far behind, nor is the patch derived from the analytic model. That is, even when we use our own model to compute our position in the relevant subspace, we perform little worse than patching the entire DAS or relevant subspace. In the *starts-with-n* case, both DAS and the analytic model perform somewhat worse, with recovery only improving around layer 23. In all cases, using a strong steering factor ($\alpha = 2$) suffices to boost recovery strongly.

F Inter-Manifold and Logit Geometry

F.1 Inter-Manifold Overlap and Orthogonality

To what degree are each of the manifolds separate? We note that Feucht et al. (2026) found that their short-period clock manifolds were orthogonal, on the basis of low cosine similarity between Fourier probes. However, our probes are trained on slightly different data, trained within our DAS subspace (instead of on the full residual stream), and include many new periods, so we perform this same analysis on our probes.

Figure 20 shows the per-probe cosine similarity at layer 18; recall that because the probes are trained within a 32D subspace, the random-chance cosine similarity is relatively high: 0.12. Most of our probe pairs are at or below random chance similarity, but we note that there are some pairs with higher cosine similarity, e.g., the probes for the sine of the fourth and cosine of the fifth fundamental frequencies of the period-1000 signal have a large negative cosine (-0.48). Positive similarities do not rise above 0.33, and in general there is little systematic similarity.

F.2 Manifold-Logit Alignment

We find that the manifolds that we study are often well aligned with the logits: an activation in the e.g. = *0 mod 2* position on the relevant manifold directly boosts the manifolds for even numbers.

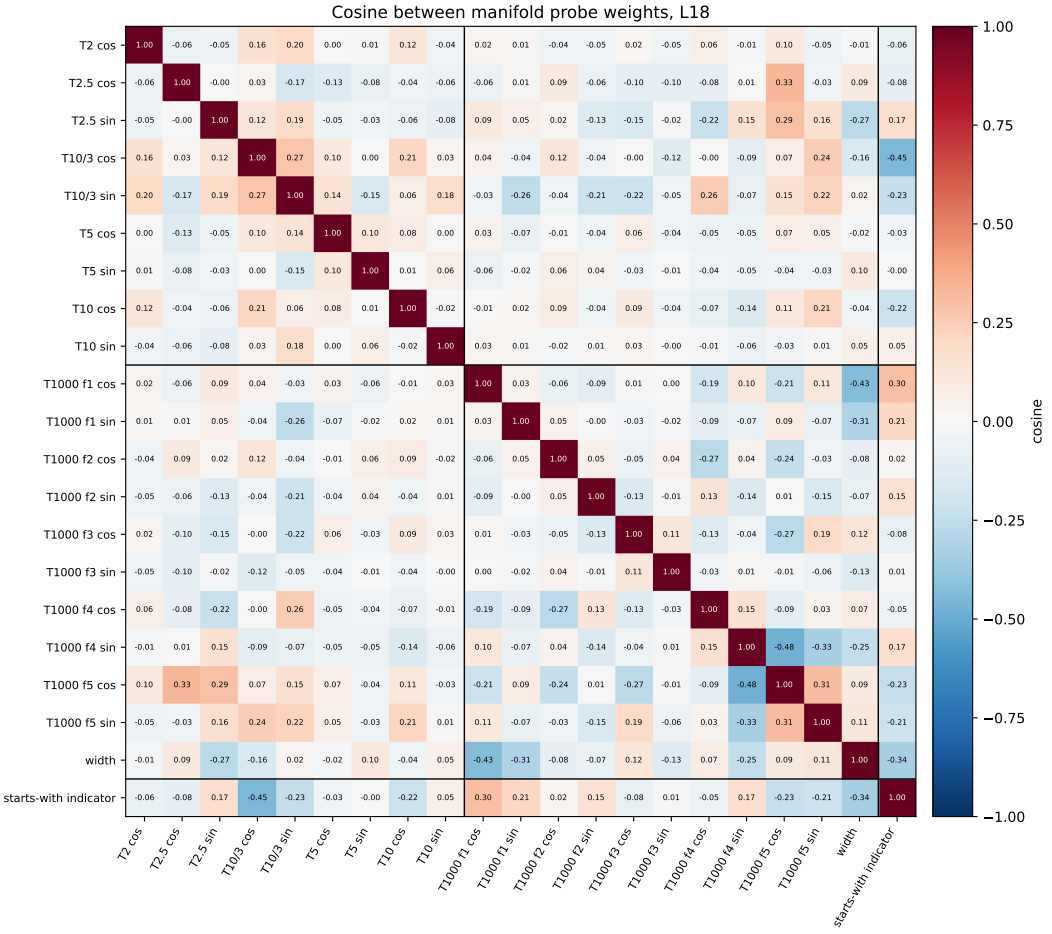


Figure 20: Cosine similarity between layer-18 probes trained to detect manifold-relevant structure.

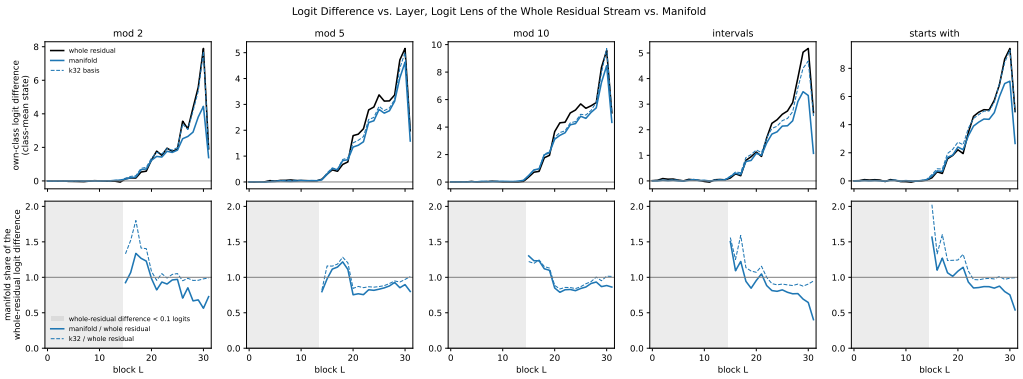


Figure 21: Per-layer mean logit difference (across h) when taking the logit lens of the whole residual, the relevant manifold, and the DAS basis, expressed as raw logit difference (top) or as a proportion of the whole-residual logit difference (bottom).

To measure this, we provide Llama with length-10 contexts drawn from a single hypothesis, and collect the output activations from every block. The hypotheses were = *n mod T* for $T = 2, 5, 10$ (960 contexts), the 400 intervals (6 contexts each), and the nine *starts-with-n* rules $n = 1 \ldots, 9$ (96 contexts each; every context contained at least one one- or two-digit member).

For a hypothesis h within a family H_f we compute $\mathbf{z}_h$, the mean state over its contexts, and center it within its manifold, yielding $\bar{\mathbf{z}}_h = \mathbf{z}_h - \frac{1}{|H_f|} \sum_{h' \in H_f} \mathbf{z}_{h'}$. We define the logit difference for h as:

$$\text{logit-diff}_h(\mathbf{x}) = \frac{1}{|h|} \sum_{n \in h} \text{logit}_n(\mathbf{x}) - \frac{1}{999 - |h|} \sum_{n \notin h} \text{logit}_n(\mathbf{x}),$$

where $\text{logit}_n(\mathbf{x})$ is the logit assigned to digit $n \in 1, \ldots, 999$ by the logit lens (nostalgebraist, 2020), with the RMS-norm applied prior to the unembedding matrix.

We compute logit-diff($\mathbf{x}$) for three choices of $\mathbf{x}$, each a median over the manifold's classes: the whole residual ($\mathbf{x} = \bar{\mathbf{z}}_h$), the manifold ($\mathbf{x} = P\bar{\mathbf{z}}_h$, with P the orthogonal projector onto the manifold's probe directions inside the DAS basis), and the full DAS basis ($\mathbf{x} = Q\bar{\mathbf{z}}_h$, with Q the orthogonal projector onto the whole DAS basis). We can express either of the latter two as fractions of the first quantity. Using the centered quantity $\bar{\mathbf{z}}_h$ allows us to study the effect of this specific hypothesis h over all of the other hypotheses H_f on the same manifold.

Comparing these three logit-diffs (Figure 21) indicates that the manifolds we define indeed capture a large portion of the logit difference. In particular, the manifold's fraction of the whole residual's logit difference is ≥ 0.8, across many layers, and most manifolds. The main exceptions to these are the intervals and *starts-with-n* at late layers, which see a drop in the amount of logit-lens mass explained by the manifold. The *mod 2* manifold is similarly a worse explanation at the end of the model.

Notably, this is not due to logits being expressed outside of the DAS subspace. In fact, the DAS subspace consistently captures nearly all of the logit difference. In the case of the *mod 2* manifold, we can train a probe for the remaining *mod 2* signal in the DAS subspace that is otherwise missed by main probe / manifold.

G Matryoshka Mask Learning

Matryoshka mask learning (Arora et al., 2026) is a technique that, given a model composed of components C, attempts to identify important components for a given task. Like past mask learning techniques, it does so by learning scores $S \in \mathbb{R}^{|C|}$ that represent how important each component is; the top-c components by score are taken to be the most important ones. However, Matryoshka mask learning differs from other mask learning methods in producing scores that reproduce model behavior well across values of c, $1 \leq c \leq |C|$. This contrasts with other mask learning methods (Davies

et al., 2023), which learn a fixed-sparsity mask, and whose scores and masks are thus unreliable outside a narrow band of c values.

Mask learning, like other localization methods, can be used to localize important components over various components, and using various ablations. Here, we choose to localize attention heads and MLP neurons separately, simply because the two exist at different scales—models have many neurons and relatively few attention heads. This means that e.g. any neurons in an attention head circuit are left as untouched mediator variables. We also learn per-position-type masks to yield finer-grained insights into which components matter where. But different components (whole MLPs, residual streams) or positions (position-agnostic or per-position masks) are also possible.

A more important question is how we ablate the masked or kept units. If one ablates the masked units, and does not intervene on kept units, one must find the sufficient set of units—from embeddings to logits—for preserving the unablated signal. This corresponds to finding a faithful circuit, in the component circuits literature. By contrast, if one ablates all masked neurons, but patches in clean outputs for the kept units, this corresponds to only finding those units with the strongest direct effect on the loss. We mostly search for circuits of the former type, but occasionally use the latter. Note that many other such ablations, corresponding to other causal relationships, are possible.

The type of ablation performed also matters. We avoid zero ablations, due to their potential for bringing model activation out of distribution (Chan et al., 2022). However, mean ablations and activation patching both largely avoid this problem. We prefer mean ablations (over the Number Game inputs), because we seek to learn per-rule masks; activation patching can create masks that mix rules. To see this, consider two neurons, one that writes to *multiples of 4*, and another that writes to *numbers that end in 1*; ideally, a *multiples of 4* mask contains the former but not the latter. If we mean ablate unselected neurons, we get this result. However, if we activation patch unselected neurons using an example from *ends-in-1*, we may end up selecting the latter too, simply because if it is ablated, it will activate when it should not.

We can also consider the loss used to train the mask. Initially, we learned these masks using KL divergence between model and circuit performance. However this finds many units responsible for the distribution of probability mass within on-rule numbers, especially in the neuron context; we would prefer to find units that decide between rules. Training on raw $p(Y \in C|X)$ alone is also possible, though circuit performance tends to overshoot that of the original model. As a result, when training to recover $p(Y \in C|X)$, we instead train to minimize the mean squared error between the circuit and original $p(Y \in C|X)$, rather than simply maximizing $p(Y \in C|X)$.

H Belief Computation Circuit Details

To identify the circuit, or sparse set of components, responsible for constructing Llama's beliefs over hypotheses, we use Matryoshka mask learning (Appendix G). Matryoshka mask learning attempts to find a ranking of components C that maximally preserves model behavior when all but the top-c components are ablated, for $1 \leq c \leq |C|$. We perform mean ablation: the output of each unselected component is set to that component's mean output at the same position, computed over a reference set of Number Game inputs described below.

We find circuits composed of attention heads, which we group based on the token at their query position: either a number, comma, or space. That is, when learning the mask, we treat "attention head A0.H0 at a number position" as a distinct entity from the same head at a space position. We do this because different heads are likely at work at different positions. This yields $3 \times n_{heads} \times n_{layers} = 3 \times 32 \times 32 = 3072$ distinct units.

We learn one mask per manifold that we find in Llama, i.e. mod 2, mod 5, mod 10, multiples of 3/9, multiples of 2/4/8/16, intervals, and *starts-with-n*. Like before, we use the pre-defined roster of 400 intervals from the behavioral evaluations, rather than the entire space of intervals.

We train the mask to minimize the sum of two squared error terms: the difference between Llama's and the circuit's probability mass on h ($p(Y \in C|X)$), and the same difference, but for numbers outside h. We measure this at each space position of our length-20 inputs. We train each mask on 960 length-20 inputs, spread evenly over the manifold's hypotheses, and evaluate it on 240 held-out inputs. The mean used for mean ablation is taken over 96 inputs from the other six manifolds.

We note that all MLP neurons are implicitly included in these circuits; they are always "on". That is, they take on their normal values even as attention heads are ablated; they act as silent mediators. We also found heads directly over these neurons, but we omit these from the present work.

Results Matryoshka mask learning finds relatively sparse sets of heads supporting each manifold that we consider. To recover 90% of model performance as measured by $p(Y \in C|X)$, relatively few components suffice: a median of 9 attention heads for each hypothesis. Moreover, there is a clear pattern to their distribution: almost all components are located either in lower-mid layers (1-14) at the number position, or in mid-high layers (13-31) at the space position. We take these 90% faithful circuits as our objects of study, and describe them briefly below.

We categorize our circuit's attention heads by the positions they attend from and to. Because a head can attend to a position without that position being causally important to the output, we first learn a position-wise Matryoshka mask over each head's values. We find that the masks naturally learn to keep each head's values at either space or number positions, almost never both.

We now show the circuits for the period-2 and period-5 manifolds (Figure 22) period-T, 2-adic, and 3-adic manifolds (Figure 23); and *intervals* and *starts-with* manifolds (Figure 24). Of note is that these circuits are relatively small, considering the size of the model in which they are found. Moreover, they contain many of the same units. For instance, at the number position, A5.H22 and A7.H25 recur heavily. These are units involved in comparison of the current number with previous numbers. Various layer 15 heads—A15.H1, A15.H2, A15.H16 and A15.H17—attend to number positions but are implicated in transferring information relevant to current vs. previous number comparison to the space position, rather than computing averages of input features; they fail both the LEACE and pullback probe tests (Appendix I).

By contrast, the majority of units in these circuits are ones that we believe perform averaging, as they attend back broadly to number positions, and pass our LEACE and pullback probe tests: A13.H4, A13.H6, A15.H19, A16.H3, A16.H8, A16.H21, A17.H24, A18.H4, A18.H18, A19.H0, A20.H2, A22.H13, A22.H14, A23.H13, A23.H14, A23.H15, A26.H13, A27.H20, A27.H6, A29.H20, A30.H16, A31.H20. Note these heads cluster somewhat by head index within a layer, which is a natural result of Llama's grouped query attention (Ainslie et al., 2023); groups of 4 heads (e.g. A0.H0-A0.H3) share a key and value, and may for this reason play related roles.

I Number-Position Information

I.1 Probing for Number-Position Information

To probe for relevant features at the number position, showing that they are available for attention heads to access, we do the following. We create 2000 comma-and-space-separated sequences of 10 numbers drawn uniformly with replacement from $[1, 999]$. At each number position, we record the full residual-stream activations at the embeddings and at each layer output.

For each representation, we predict various functions of the number x from which it was taken. Relevant for the *multiples-of-n* and *ends-in-n* rules, we predict $\cos$ and $\sin(2\pi x/T)$ for $T = 10, 5, 10/3, 5/2$ and 2; we do not predict the sin at period 2, as it is zero on all integers. We also predict binary indicators for whether x is divisible by 2, 4, 8, or 16, and whether x is divisible by 3 or 9. As controls, we also predict Fourier features for $T = 3, 4$ and *divisible-by-n* for $n = 7, 11$

Relevant for the intervals, we predict $\cos$ and $\sin(2f\pi x/1000)$ for $f = 1, \ldots, 5$, i.e. periods 1000, 500, 333, 250 and 200. Finally, relevant for *starts-with-n*, we predict an indicator, $x < 100$, as well as the leading digit of x as a nine-way classification.

We fit the Fourier probes with least squares, while the indicators are class-balanced logistic probes, and the leading digit is a class-balanced softmax probe; both of the latter use L2 regularization. We split the numbers 1-999 in an 80/20 train/test split, testing probes on unseen numbers. We measure R2 for the regression targets, area under the ROC curve for the indicators, and accuracy for the leading digit.

Our results (Figure 25) suggest that the features relevant for our manifolds are all present at the number position. Almost all of the relevant features are present right from the embeddings; the only

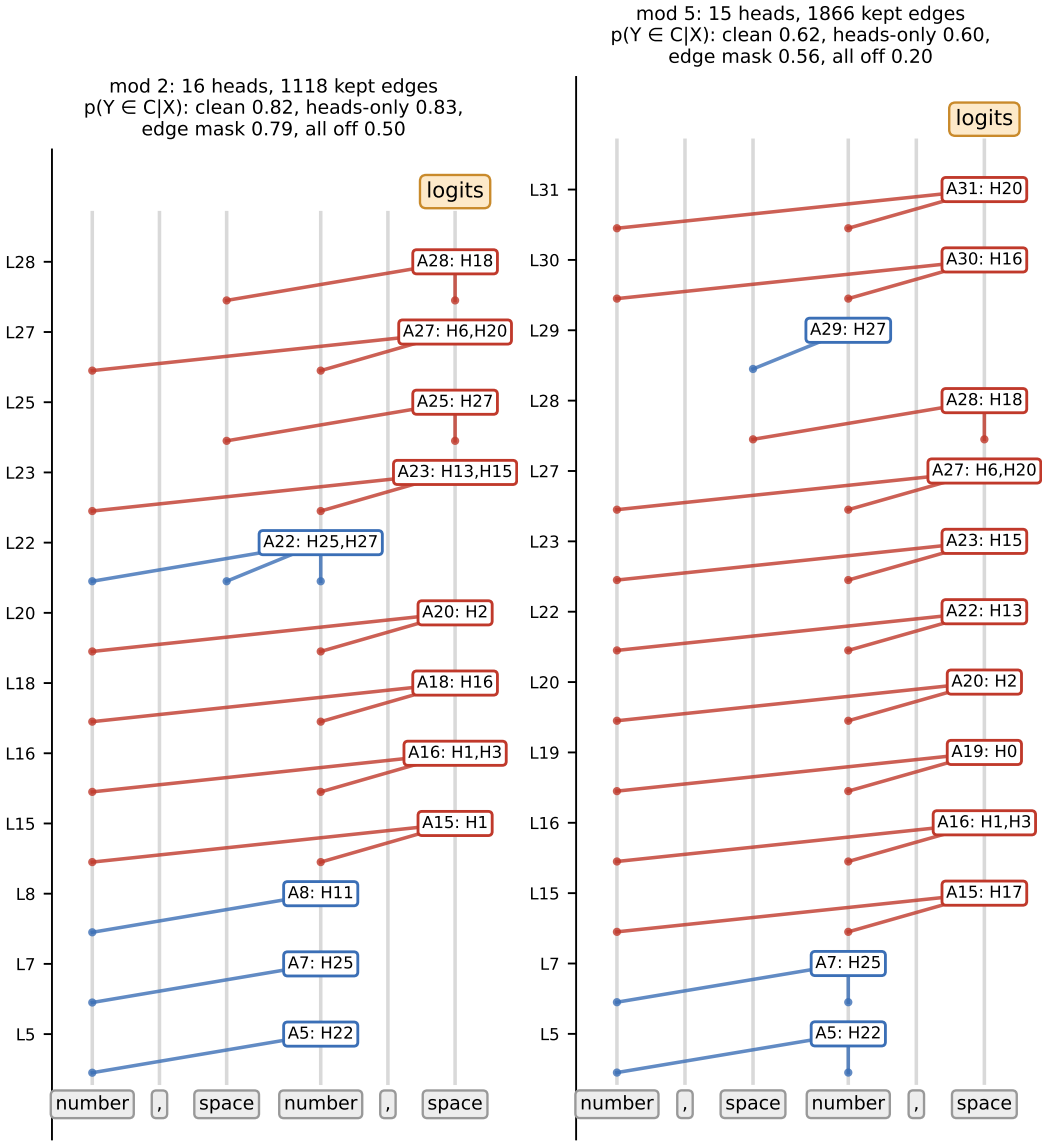


Figure 22: Llama's circuits for the mod-2 and mod-5 manifolds / hypotheses.

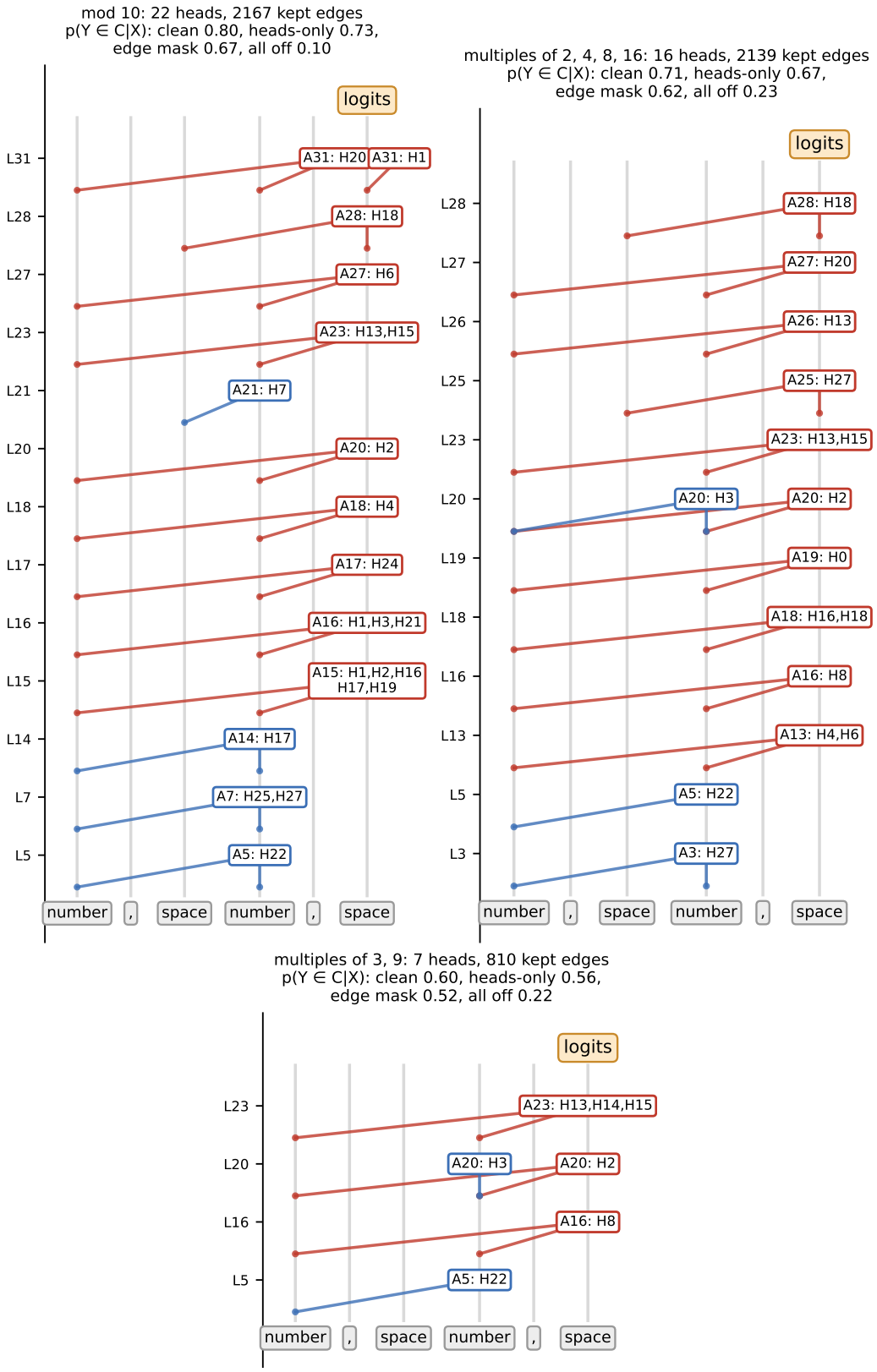


Figure 23: Llama's circuits for the mod-10 manifold, and the 2-adic (number of times 2 appears as a factor) and 3-adic manifolds

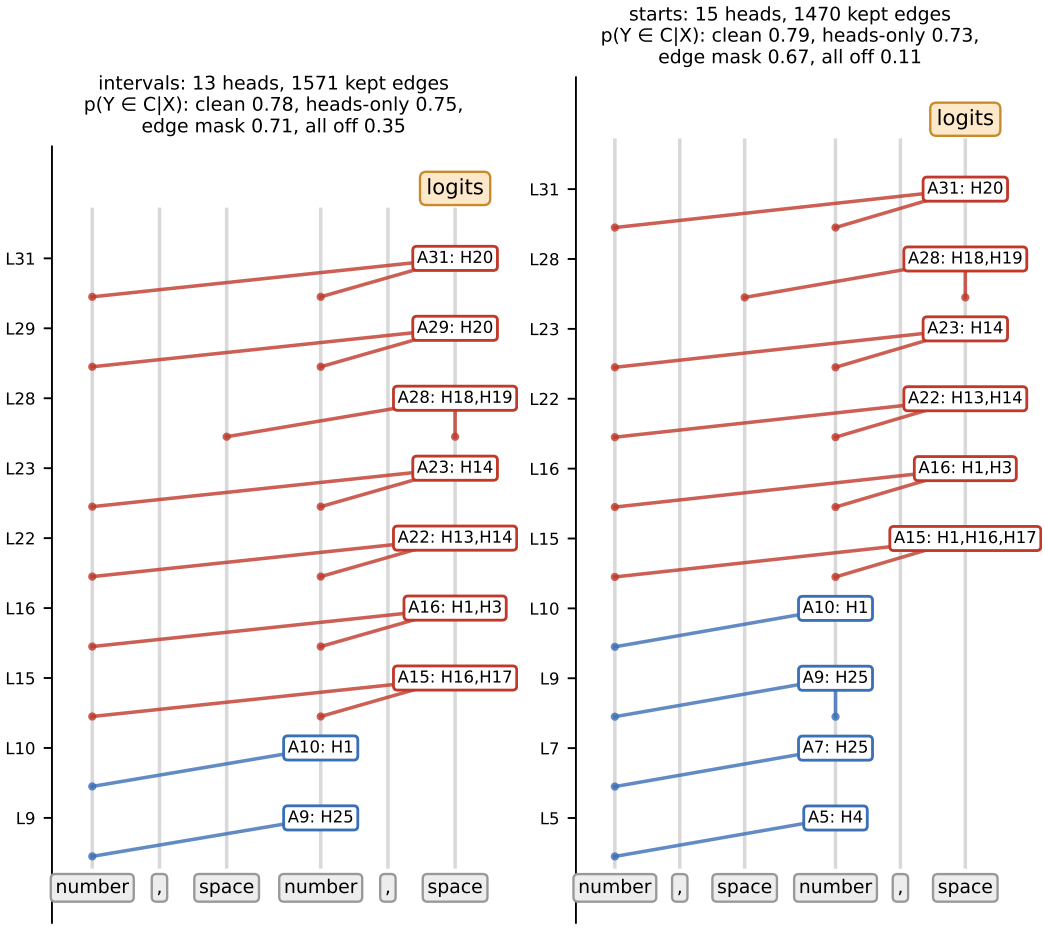


Figure 24: Llama's circuits for intervals and *starts-with-n* manifolds / hypotheses

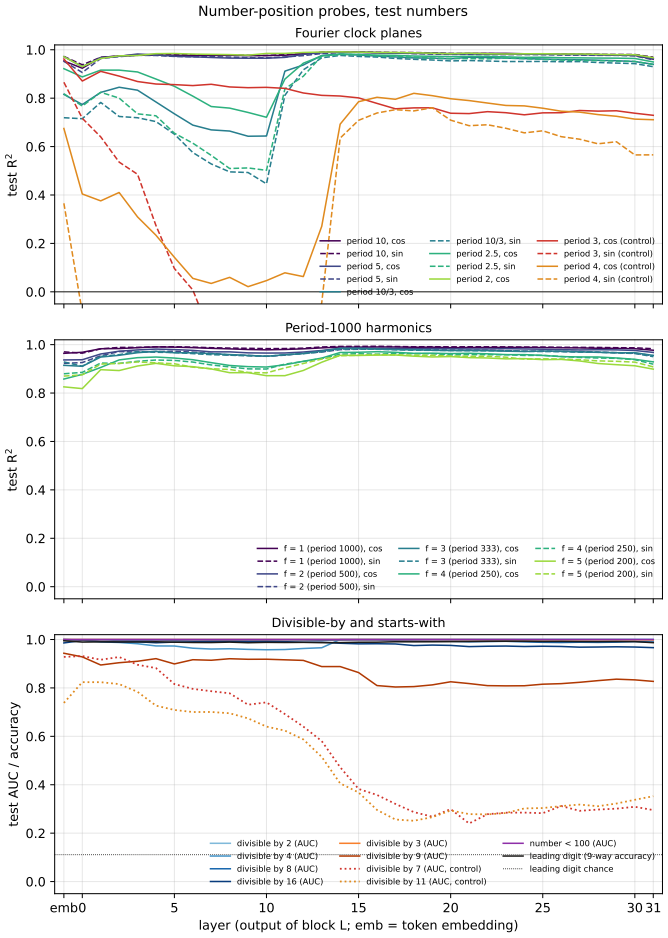


Figure 25: Accuracy of various probes trained at the number position.

exceptions are the higher-order harmonics of period-5 ($T = 5/2$) and period-10 ($T = 10/3$), which only emerge closer to layer 12. The controls, by contrast, achieve worse performance, with huge dips in R^2 before recoveries around layer 15, in the case of the period-4 probe, and no recovery for period-3. The *divisible-by-n* controls see their performance drop over the layers, settling at an AUC around 0.3 by layer 18.

I.2 LEACE Experiments

To test whether the heads we believe to be averaging (here, in-circuit heads at the space position) read the manifold features from the number positions, we erase those features from one head's input at a time, using LEACE (Belrose et al., 2023). LEACE removes all linearly-decodable information about a certain feature from a representation, at least on the data it is fit to. So if removing relevant features with LEACE causes the same amount of damage as a mean ablation, we know that the head was using or transmitting those features.

The information that we remove using LEACE varies based on the manifold in question: for the mod 2/5/10 manifolds, we remove the relevant short-period Fourier features; for the intervals, we remove the first 5 harmonics of the period-1000 Fourier features. For starts, we remove information about the leading digit (encoded as 9 binary indicators, one per digit), as well as a binary indicator for $n < 100$. We also remove the divisibility indicators (i.e., a binary indicator for divisibility by 2/4/8/16 or 3/9) for the power families.

We fit LEACE at each layer of the residual stream that tested heads take as input, at every number token of 576 contexts. These contexts come from the other six manifolds, 96 per manifold, so that we do not train on the manifold that we apply LEACE to. After LEACE is trained, we take the head we wish to test and recompute only this target head's values from the erased residual. The attention pattern and all other heads stay clean. As a comparison, we replace the head's number-key values with their mean. We measure the drop in $p(Y \in C|X)$. When we do this normally, effects can be hard to see, as intervening on one individual head does little; other heads can compensate. When we do this in the context of the belief construction circuit (i.e. while ablating all non-circuit heads), there is less redundancy and the effects are more visible. For this reason, we report the in-circuit LEACE results.

For most heads, LEACE matches mean ablation to within 0.04. The largest single-head drops stem from A20.H2. Erasing the features from all heads at once removes 0.20–0.42 of $p(Y \in C|X)$, close to mean ablation (0.23–0.53). The main exceptions are the A15 heads and A30.H16. For these heads, LEACE has almost no effect, but mean values do (e.g. *starts-with-n*, A15.H16: 0.00 vs 0.03). Their values carry number content other than these features. Note that some heads show no LEACE effects but also no mean ablation effects, perhaps because it is the distribution of attention, rather than what they attend to that matters.

I.3 Pullback Probes

In addition to probing directly for manifold-relevant information at the number position, we use a semi-causal method to probe for the presence of such information. Consider an in-circuit attention head that we believe to be computing an average of manifold-relevant features of the input numbers. If it is in the circuit for a given manifold, we take the pullback of each manifold probe on the residual stream right after the head's attention layer (before the MLP). If we pull this gradient back through this head's values (with frozen attention patterns), to its inputs, we obtain a direction that, when written to, increases the probe's readout at the head's output. This direction can be used as a probe for the number-level analogue of the manifold-level feature.

Concretely, for the circuits of mod 2, mod 5, mod 10, and intervals, we take all of the heads whose attention from spaces goes mostly to numbers (ignoring BOS). We then take the pullback of each (space-position) probe for that manifold through that head, summed over number positions and contexts, and use it to predict the same Fourier feature of the number (e.g. $\cos(2\pi n/10)$) from number-position activations at the head's inputs, computing held-out R^2.

Our results (Figure 27) are surprisingly good. Only 5 heads fail to provide good probes when used in this way; 4 of them are known to be heads involved in computing interval width (which does

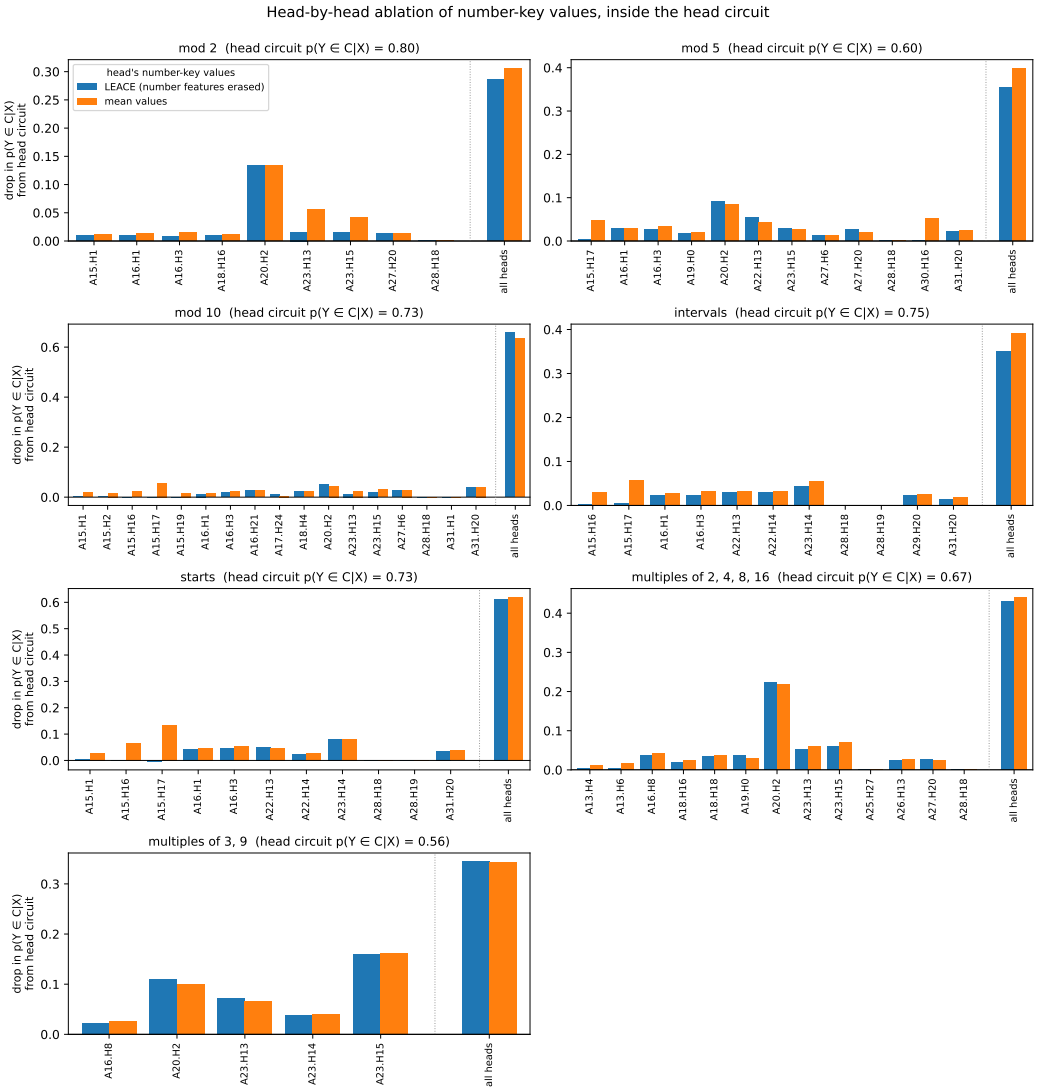


Figure 26: Results of performing LEACE on in-circuit heads' value vectors while non-circuit heads are ablated, compared to mean ablating the value vectors. Orange shows mean ablation effects, while blue shows LEACE effects; if the two bars are similar, the head is likely transmitting number features.



Figure 27: AUC of the pullback of the space-position probes through the values of each individual space-position, number-attending head in the relevant manifold's circuit. The pullback via these heads is a surprisingly effective probe; the L15 heads that fail compute interval width, and do not aggregate per-number features.

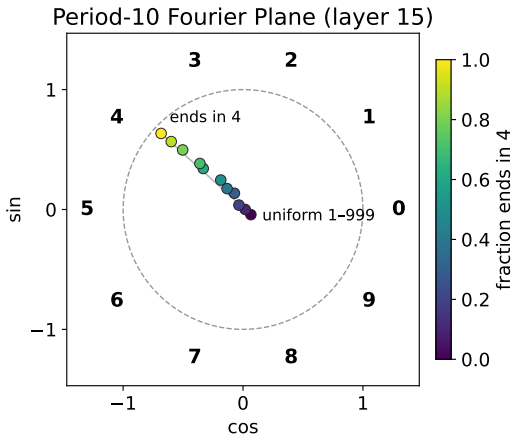


Figure 28: Given mixed evidence, we do not see exponential scaling of on-manifold norm at layer 15, unlike in Figure 5 (center). This is likely constructed at later layers.

not have a corresponding number-level feature, as it involves comparing multiple numbers). This suggests that the heads are indeed constructing these manifolds.

J A Context-Consistency Circuit

Our search for context-consistency circuits is motivated by the observation that a) not all heads in the belief-construction circuit seem to participate in averaging, as measured by the LEACE and pullback probe experiments, and b) that the shrinkage of mixed-consistency inputs is likely caused by something that is not merely its position on the plane. Consider Figure 28, which shows the same inputs as were shown in Figure 5 (center), at layer 15; only in later layers do we see exponential scaling with f. So, there is likely a circuit responsible for this.

We learn one attention-head mask per hypothesis family: residues mod 2, mod 5 and mod 10; intervals, *starts-with-n*; multiples of 2, 4, 8 and 16; and multiples of 3 and 9. For each family we sample contexts of 20 numbers. Each context draws its numbers from a hypothesis within the relevant family. We use 960 training contexts and 240 held-out contexts per family.

We split the 20 number positions of each context at random into two equal-sized sets, A and B. We build two copies of the context: X_A keeps the original numbers at A and replaces the numbers at B with numbers drawn uniformly from outside of the concept C. X_B does the reverse. Thus, each number in X has a corresponding counterpart in X_A or X_B that has the same number identity, but sits in a context that is only 50% drawn from C.

When we ablate a head, what we change depends on its query position. At a number position j, we replace the head's output with its output at the same number token in X_A if j is in A, and in X_B if j is in B. At comma and space positions, the head keeps its attention pattern, but at each number position j it reads the value vector computed at the same number token in X_A if j is in A, and in X_B if j is in B. Values at other positions do not change. Llama-3.1 (8B) uses grouped query attention, and thus shares each value head across four query heads; however, we apply masking per query head.

This procedure ensures that two groups of heads are captured. First, heads at the number position that gather contextual information from other number positions are found, because we ablate the contextual information they gather. Second, heads at space or comma positions that use this contextual information are also found, because we ablate the contextual information they read in. Any heads that use number-identity information or functions thereof are ignored by the mask, because our ablation preserves the current number's identity at the number position.

We learn masks per head and position type (number, comma, space), as in prior sections. We train them using Matryoshka mask learning with the same objective as used previously: at every space

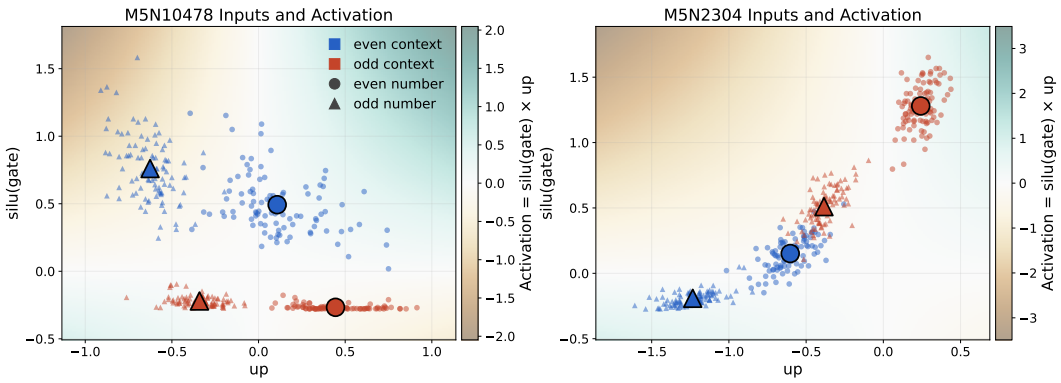


Figure 29: Activations of two parity matching neurons. We vary both the context (even / odd) and current numbers (even / odd), and plot the neurons' up and gate projections, which are multiplied together to get the neuron's activation (the background color). The neurons clearly utilize both of their inputs in order to compute parity match, equivalent to the XNOR operation.

position we compare the masked model with the clean model on two quantities: $p(Y \in C|X)$, and the probability of numbers outside C. The loss is the squared error of both, averaged over positions and contexts. We train for 4000 steps with Adam (our learning rate is 0.05, and our batch size is 24).

After training, we rank heads using the scores produced by Matryoshka mask learning. For each k we keep the top k heads, ablate all others, and measure pooled $p(Y \in C|X)$ on the held-out contexts. We report the smallest k that recovers 90% of the gap between the fully ablated and the clean model. We also repeat this same procedure for number-position MLP neurons; in this case we simply replace their output with their output on X_A or X_B, whichever is appropriate.

The attention circuits we find are small and share many heads among them: they contain 17–32 head-position units each (15–29 distinct heads), and 73 unique heads in total. Of the 20 heads shared by at least three consistency circuits, 13 also appear in the belief-construction circuits. Six of these are number-position heads: A5.H22, A7.H25, A7.H27, and A9.H25, which attend back to number positions, and A22.H25 and A22.H27, which attend to spaces. The other seven are space-position heads, containing precisely the group of A15 heads that do not perform averaging: A15.H1, A15.H2, A15.H16, A15.H17 and A15.H19. Only A23.H13 and A23.H15 appeared to perform averaging in our LEACE and pullback tests. In addition to these, there are a variety of new heads that do not appear in our previous circuits.

What do these heads do? Since the lowest-level heads look back at other numbers, we postulate that they are computing some function of the context. But extracting semantics from heads is hard, so we instead find neuron circuits using the same procedure as above. We only mask number-position neurons, to find what exactly is being computed by the number-position heads (and likely transmitted by the L15 heads). Using Matryoshka mask learning, we also find relatively small neuron circuits: they range from 8-229 neurons in size, given 14336 neurons per single MLP.

We focus here on characterizing the neurons behind the period-2, 5, and 10 Fourier planes. We start with the period-2 Fourier plane, and consider its top neurons. We believe them to be computing context consistency, so we measure their activations when all prior numbers share a parity, and the current number (at which the neuron is located) either shares or does not share that parity. A neuron whose activation separates matches from mismatches is sensitive to parity.

Over half of the top-10 neurons in this circuit are sensitive to parity matching by this measure. The top neurons in the circuit (#1 and #3) are computing the parity, while others downstream appear simply to relay it; we can distinguish this by asking whether match and mismatch are already linearly separable at the neuron's inputs, or only at its activation, suggesting the neuron had to compute it. We show how the computing neurons compute parity match in Figure 29. Recall that Llama's MLP neurons have two inputs, $W_{up}^{\top}\mathbf{x}$ and $\mathrm{Swish}(W_{gate}^{\top}\mathbf{x})$, which they multiply together to compute their activation. In order to compute the parity match operation, an XNOR of context and current number parity, these neurons use both of their inputs. Note that both of these neurons are in layer

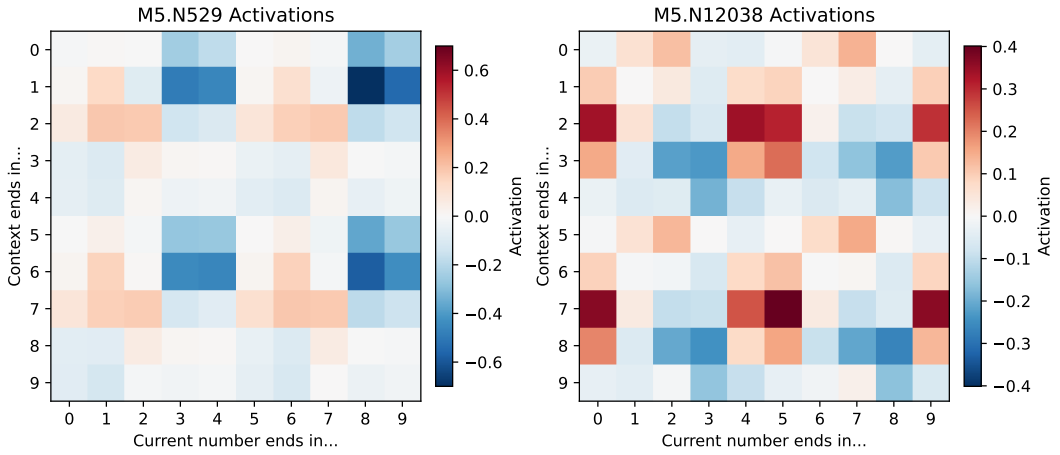


Figure 30: Activations of the mod-5 consistency-checking circuit as we alter the number that the context ends in, and that the current number ends in. The two neurons compute different, complementary matches, and show clear quadrant-wise periodic structure.

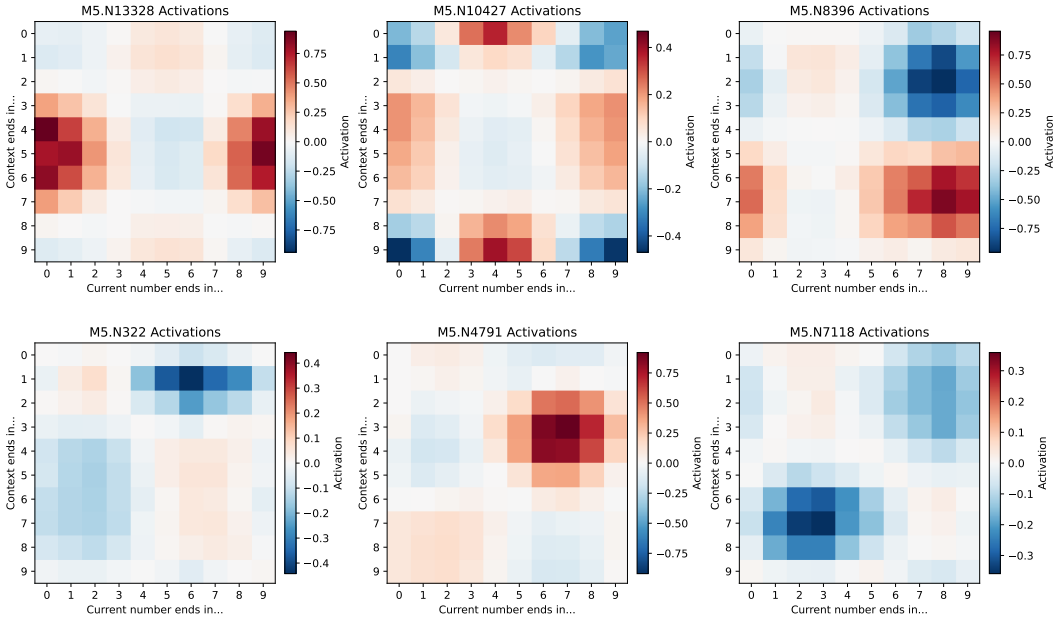


Figure 31: Activations of the mod-10 consistency-checking circuit as we alter the number that the context ends in, and that the current number ends in. The six neurons jointly tile the space of end-digit matches.

5, suggesting that they are fed by the upstream in-circuit head A5.H22. And indeed, if we take the pullback (through A5.H22) of each neuron's gate projection, which for both neurons computes context parity, this also classifies the context numbers' parity well. That is to say, that A5.H22 transmits the average parity of the context to these neurons, which they use to compute parity.

We then perform analogous experiments for the period-5 and 10 Fourier planes. We similarly identify neurons that relay matching information and that compute them. However, the computation of matching information is more distributed than in the mod-2 case. While the mod-2 consistency neurons compute essentially the same thing twice, the mod-5 consistency circuit consists of neurons that essentially tile the space, checking for consistency for different contexts and current numbers.

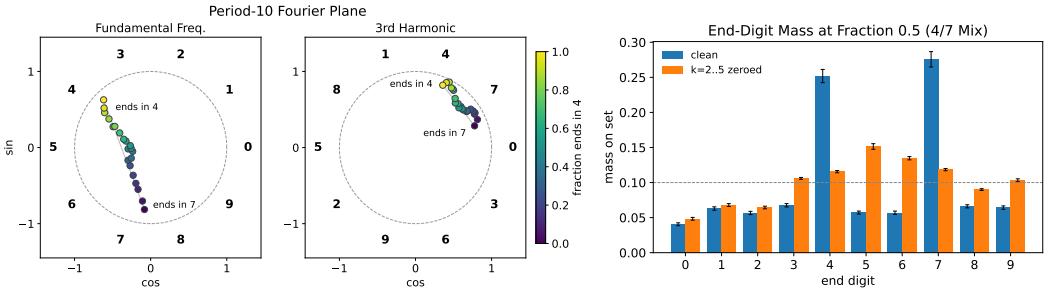


Figure 32: **Left**: Llama's representations of mixed *ends-in-4/7* input on the original period-10 Fourier plane. As we alter the fraction of *ends-in-4* numbers, the representations walk the chord from position 4 to 7, leading to representations that look like *ends-in-5/6*. **Center**: Llama also represents numbers with respect to the third harmonic of mod 10; in this representation, the *ends-in-4/7* are neighbors, and there is no interference from *ends-in-5/6*. **Right**: Given equal proportions of *ends-in-4/7* in its input, Llama prefers numbers ending in 4/7 , due to the third-harmonic representation (blue). If non-fundamental frequencies are ablated, a weak *ends-in-5/6* preference appears (orange).

Figure 30 shows how this is performed, displaying the neuron's activations based on the current ending number and the context's ending number. We can see that each quadrant of activations is almost identical—precisely what we'd expect from a period-5 neuron.

The mod-10 consistency computation circuit is similar in character to mod 5, but yet more complex. Figure 31 shows the six neurons we found to be involved, which also tile the space and have no quadrant-wise structure, as expected for period-10 neurons. This recapitulates patterns reported by Feucht et al. (2026), *inter alia*, whereby consistent semantic patterns (matches between ending digits) are spread across neurons whose activations and writes tile the space.

Based on the neurons circuits that we have found, we hypothesize that many of the remaining neurons serve simply to reinforce, gate, or simply relay the patterns computed by these neurons, and that downstream, the non-averaging heads transmit this information to the space position.

However, we stress that this is only an initial characterization of these neurons and circuits. Although the neuron circuits we find are sparse, we do not interpret all neurons. And, while we believe that similar mechanisms are at play for other rules (in particular, the parity circuit can also deal with other multiplicities of 2), we do not show this, or show that these circuits are directly responsible for the exponential increase in logits to a given hypothesis as the data match it better. What we can say is that Llama has additional, non-averaging machinery that influences their outputs. This seems to affect the norm of the representation, rather than where it lies on the manifold (as the heads involved here are not averagers).

K Maintaining Performance Under Uncertainty

K.1 *Ends-in-n*

In this section, we explain how Llama's redundant Fourier representations allow it to maintain performance under uncertainty. We give Llama an input that mixes 20 numbers that end in 4 and in 7. In Figure 32 (left), we show how Llama's representations in such scenarios change as we change the ratio of *ends-in-4* to *ends-in-7* numbers. The representations walk the chord between the 4 and 7 positions on the period-10 Fourier plane, clearly reflecting the ratio in the input. We can do the same with e.g. *starts-with-2* and *starts-with-6*.

Surprisingly, this does not lead to the ambiguous behavior suggested by these representations: a mix of *ends-in-4/7* inputs still yields primarily *ends-in-4/7* outputs. This is not because the representational geometry is causally irrelevant; it is because the geometry is robust to ambiguity. Llama represents numbers and rules not only via their first fundamental frequencies mod T but also their higher-order harmonics: probing for $\cos$ or $\sin\left(\frac{2f\pi x}{T}\right)$ for $f = 3, 4$ yields high accuracy at the

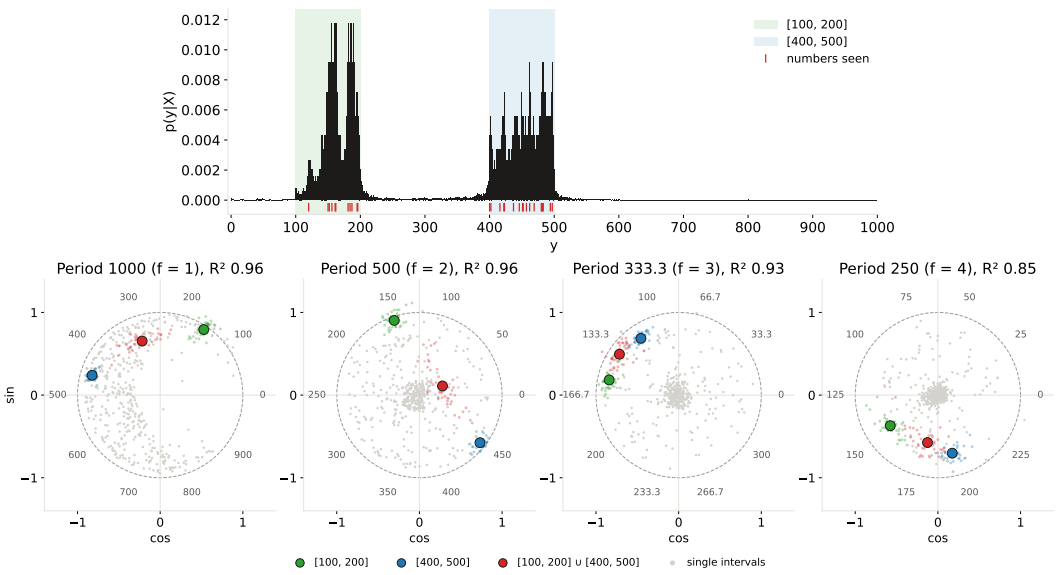


Figure 33: **Top**: Llama's output when given input that is sampled both from the interval [100,200] and from [400,500]. Its output reflects this bimodal distribution. **Bottom**: While Llama's representations on mixed input are very ambiguous with respect to the first fundamental frequency, they are much less ambiguous with respect to other higher-order harmonics.

same layers where we found the original ($f = 1$) plane ($f = 2, 5$ are the period-5 and 2 planes). Past work has found higher-order harmonics in toy models (Nanda et al., 2023; Zhong et al., 2023), and suggested they might exist in larger models as well (Štefánik et al., 2026).

The representation of a mixed *ends-in-4/7* inputs does not resemble a low-norm *ends-in-5/6* representation at harmonics beyond the fundamental frequency. At the 3rd harmonic, the *ends-in-4/7* representation is ambiguous only between *ends-in-4/7* (Figure 4, center); because the two are adjacent on the plane, this harmonic also has the highest norm and dominates the model's behavior. If we zero-ablate the higher harmonic planes, the ambiguous representation weakly dominates, and *ends-in-5/6* receive the most probability (Figure 4, right). The *starts-in* curve similarly has added structure in higher dimensions that reduces interference in ambiguous settings, discussed in Appendix D.

K.2 Intervals

A similar mechanism can be found at work in the interval case. Consider the case of a length-30 input mixing two intervals: [100,200] and [400,500]. While these might appear to yield a single interval at [250,350] in the first fundamental frequency of 1000, at other harmonics, specifically the 3rd, this is not the case (Figure 33, bottom). As a result, Llama is able to act on this ambiguity behaviorally (Figure 33, top).

L Evidence Accumulation and Counting

Here, we discuss the experiments related to evidence accumulation, including the precise experimental setups used and results obtained. Throughout this section, we consider the seven manifolds of the main text (mod 2, mod 5, mod 10, intervals, starts-with, multiples of 2/4/8/16 and the multiples of 3/9) and their hypothesis sets, and use $p(Y \in C|X)$ as our measure of model performance.

L.1 MLP 31 WRITES MORE STRONGLY AS N GROWS

For Figure 6 (left), we measure the direct write of MLP 31 toward h as:

$$\Delta_{31}(X) = \frac{1}{|h|}\sum_{y\in h}\ell_{31}(y) - \frac{1}{|\bar{h}|}\sum_{y\notin h}\ell_{31}(y), \qquad \ell_{31}(y) = W_U[y]^\top(\gamma \odot m_{31}),$$

where m_{31} is the MLP 31's output and γ is the gain parameter of the final RMSNorm. We use 128 contexts per hypothesis and read every space after example $N = 1, \ldots, 30$.

We finally note that while MLP 31 is the strongest writer for many hypotheses, this is not the case for all, e.g. *multiples-of-3*, which is supported by MLP 30.

Circuit discovery. For circuit discovery, each input X contains 15 numbers drawn from one of the aforementioned hypotheses associated with a manifold; during training we cycle through hypotheses (e.g. between the different residuals mod 5). For example, for the hypothesis $x \bmod 5 = 1$ we draw

X = `[BOS]471, 86, 766, 21, 481, 951, 341, 636, 76, 451, 111, 101, 756, 126, 751,`

and split it into three inputs of five numbers each:

$X_{1:5}$ = `[BOS]471, 86, 766, 21, 481,` $X_{6:10}$ = `[BOS]951, 341, 636, 76, 451,`
$X_{11:15}$ = `[BOS]111, 101, 756, 126, 751,`

We treat each attention head at each position type—number tokens, commas, non-final spaces, or the final space—as a unit over which a mask can be found, leaving $4 \times 32 \times 32 = 4096$ units. We learn a mask (indicating whether or not the unit is in the circuit) for each unit. To ablate a unit, we replace its output at each token of that type in X with its output at the same token in the length-5 input that contains it: at `636`, the head's output on X is replaced by its output at `636` in $X_{6:10}$; at the final space, by its output at the final space of $X_{11:15}$. The ablated model thus reads the same 15 numbers, but each ablated head sees at most five examples.

We then learn Matryoshka masks over these 4096 units. As our training loss, we use KL divergence between the unablated and masked output distributions at the final space of X. We train one mask per manifold on 960 contexts for 4,000 Adam steps (batch size 24, learning rate 0.05, temperature 0.5).

We evaluate on 240 held-out contexts. We measure faithfulness as the fraction of the difference in $p(Y \in C \mid X)$ between the fully ablated and unablated models that the top-k units recover when all other units are ablated. A circuit is the smallest top-k set with $\geq 95\%$ faithfulness. The circuits contain 53 (*starts-with-n*) to 88 (residues mod 2) units: 18-37 at number positions and 32-50 at the final space. The 80% overlap reported in Section 5 is the fraction of each manifold's feature-averaging space heads that are also final-space heads of the corresponding evidence-accumulation circuit, averaged over manifolds.

Attention vs. values decomposition. An attention head's output is $o = \sum_j a_j v_j$, where a_j is its attention to position j and v_j is the value there. We replace (1) only the values, (2) only the attention pattern, or (3) the whole output with their versions from the length-5 input. Table 1 reports the change in $p(Y \in C \mid X)$ when we do this for all circuit heads at one position type at once, as a fraction of the circuit's recovery. At the final space, values and attention matter about equally in total, but single heads vary widely. At number positions, most heads act via their attention patterns.

Attention-spreading interventions. For each context, we take the length-15 input X and its last five numbers X', which is the same input as $X_{11:15}$. We first run X with the circuit units running normally and all other units ablated, and record each circuit head's attention pattern at every position of its unit's type. We then run X' and edit the attention of every head at the matching positions of X' (number-position heads at the five numbers, a final-space head at the final space).

We test two edits. In both, we set each head's attention to the BOS token to match its attention to BOS on X, and manipulate the remaining attention. In the first edit, the remaining attention goes to

		mod 2	mod 5	mod 10	intervals	starts	mult 2-16	mult 3,9	mean
space	vals.	$-.69$	$-.53$	$-.37$	$-.34$	$-.41$	$-.51$	$-.46$	$-.48$
	attn.	$-.29$	$-.58$	$-.41$	$-.57$	$-.45$	$-.43$	$-.47$	$-.46$
	output	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00
numbers	vals.	$-.11$	$-.12$	$-.08$	$-.07$	$-.13$	$-.10$	$-.08$	$-.10$
	attn.	$-.57$	$-.48$	$-.38$	$-.45$	$-.48$	$-.43$	$-.34$	$-.45$
	output	$-.62$	$-.59$	$-.46$	$-.49$	$-.56$	$-.50$	$-.42$	$-.52$

Table 1: Change in $p(Y \in C \mid X)$ after replacing the values, attention pattern, or output of all circuit heads at one position type with their length-5 versions, as a fraction of the circuit's recovery.

the tokens of X' in proportion to how much the head attended to the corresponding tokens of X's last five examples; we call this the *last-5-example weights* edit. In the second edit, the remaining attention goes to the tokens of X' in proportion to the head's own attention to them on X'; in Figure 6 (right), this is the *5-example weights* line. We leave unchanged those positions where the head's original attention to BOS on X' exceeds 0.999. All other heads, and all values, are computed normally on X', so an edit in one layer changes the inputs of later layers. We measure

$$\frac{p_{\text{edit}}(Y \in C \mid X') - p(Y \in C \mid X')}{p(Y \in C \mid X) - p(Y \in C \mid X')}$$

on the 240 held-out contexts of each manifold, averaged over hypotheses. We edit the 95% circuits (59-192 units), and the top 10%, 15%, 20%, 30% and 50% of all 4,096 units by the same mask ranking, and all units. With the first edit, the circuits recover 0.71 of the difference, while the top 10%, 15% and 20% of units recover 0.82, 0.90 and 0.97, and all units 1.00. With the second edit, the circuits recover 0.65, the top 10%, 15% and 20% of units 0.66, 0.69 and 0.74, and all units 0.78 (Figure 6, right).

Hyperbola vs. Power Law Which curve fits Llama's behavior better? We compare how well a hyperbolic form $aN/(N+c)+b$ fits two different quantities, as compared to the earlier-proposed power law $aN^p + b$ ($.05 \le p \le 1$).

MLP 31's write. The first quantity is MLP 31's write toward h (as in Figure 6, left), at each $N = 1, \ldots, 50$. We fit 14 hypotheses, with 128 contexts for each: residues mod 5 (0, 1, 2), $x \bmod 10 = 1$, multiples of 2 and 10, odd numbers, starts-with-2, and six intervals of width 13-630. We leave out multiples of 3, whose write is made by MLP 30.

The hyperbolic form fits better for all 14 (median R^2 0.995 vs. 0.981). The differences are small, but for many hypotheses, the power law overshoots the write beyond $N \approx 40$ (Figure 34).

Manifold amplitude. The second quantity we fit to is the position of the layer-26 representation on the hypothesis's manifold, at each $N = 1, \ldots, 30$. We do this because we believe that, if the number-averaging heads assign higher total weight to content tokens, the amplitude of their write should also grow. So, for $= n \mod k$ hypotheses, we read the Fourier probe at the hypothesis's own period and record the length of the hypothesis' embedding onto the manifold.

We fit 7 residue hypotheses (residues mod 5 (0, 1, 2), $x \bmod 10 = 1$, multiples of 2 and 10, odd numbers) and 5 intervals of width 80-630. The hyperbolic form fits better for 11 of the 12 curves (median R^2 .974 against 0.937); the exception is $x \bmod 5 = 2$ (0.988 against 0.994). The gap is largest where the amplitude levels off early, e.g. $x \bmod 10 = 1$ (0.984 vs. 0.878) and $[183, 402]$ (0.970 against 0.859) (Figure 35).

M Evidence Accumulation Beyond the Number Game

To test how evidence accumulation works outside of the Number Game, we perform experiments on the following in-context math tasks. For each task, we define a low-evidence setting (where fewer in-context examples have been given) and a high-evidence setting where more has been given. We then try to map between them using attention interventions, as seen previously.

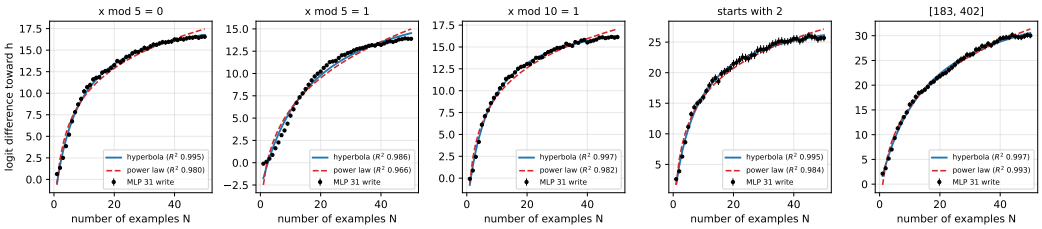


Figure 34: MLP 31's write toward h versus the number of examples N (mean over 128 contexts $\pm$ SEM), with the best hyperbolic and power-law fits, for the five hypotheses of Figure 6 (left).

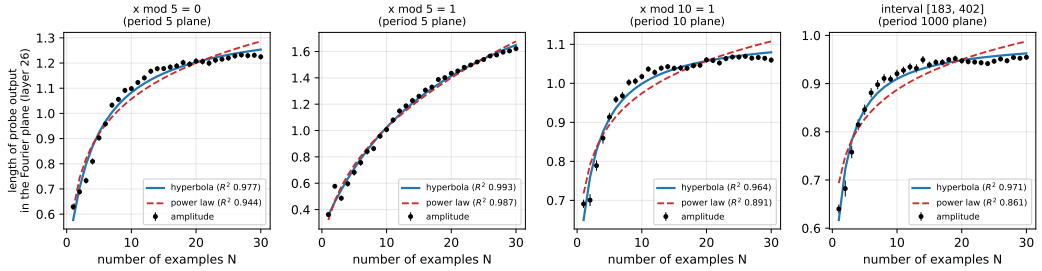


Figure 35: Amplitude of the layer-26 representation along the hypothesis's Fourier direction versus the number of examples N (mean over 128 contexts $\pm$ SEM), with the best hyperbolic and power-law fits.

- **Arrow tasks**: few-shot in-context learning tasks in the "x $\rightarrow$ y" format, comma-separated, ending with a bare query "q $\rightarrow$ ". In the high-evidence setting, we provide 15 demos and a query. The corresponding low-evidence setting consists of the last 2 demos and query of the high-evidence prompt. Inputs are drawn from 1-999 without replacement, restricted such that all outputs also lie in 1-999.
  - arrow_add: $y = x + b$, where b is drawn from 2-500. Example: 40 $\rightarrow$ 43, 37 $\rightarrow$ 40, 27 $\rightarrow$ [30]
  - arrow_subtract: $y = x - b$, where b is drawn from 2-500. Example: 35 $\rightarrow$ 33, 69 $\rightarrow$ 67, 37 $\rightarrow$ [35].
  - arrow_double: $y = 2x$. Example: 62 $\rightarrow$ 124, 73 $\rightarrow$ 146, 36 $\rightarrow$ [72].
- **Sequence tasks**: comma-separated number sequences that follow a given rule. High-evidence sequences consist of 15 terms, while corresponding low-evidence sequences consist of the last 4 terms of the high-evidence inputs.
  - sequence_ascending: ascending arithmetic sequences with a step size b drawn from 2-65. Example: 36, 38, 40, 42, [44].
  - sequence_descending: descending arithmetic sequences, with a step size b drawn from 2-9. Example: 46, 40, 34, 28, [22].
  - sequence_geometric: doubling sequence starting at a, where a is drawn from 2-7. The high-evidence setting is capped at 7 terms, since all terms must remain below 1000 for them to be single tokens; $a \cdot 2^7 \leq 999$ maintains this desideratum. Example: 56, 112, 224, 448, [896].
- **Addition**: comma-separated addition equations like "$a + b = c$" with a drawn from 1-998 and b from 1 to $999 - a$, ending with an unsolved query "$a + b =$". Because the model becomes confident quickly on such tasks, we provide only 1 in-context example in the low-evidence setting, but 15 in the high-evidence setting: 54 + 2 = 56, 68 + 6 = [74].

We draw 64 prompt pairs per task, where each low-evidence prompt consists of the final tokens of its paired high-evidence prompt. Given this data, we test whether moving attention off the BOS token is enough to shift low-evidence behavior toward high-evidence behavior, using two interventions.

			% recovery	
Task	Low evidence	High evidence	BOS only	Copied pattern
arrow_add	0.062	0.653	20%	42%
arrow_subtract	0.024	0.508	15%	37%
arrow_double	0.656	0.995	75%	63%
sequence_ascending	0.587	0.976	92%	25%
sequence_descending	0.751	0.966	107%	75%
sequence_geometric	0.825	0.945	120%	72%
addition	0.921	0.994	−5%	49%

Table 2: Llama behavior on non-Number-Game in-context math tasks. The probability assigned to the correct answer increases with evidence. Both attention edits increase the probability that Llama assigns to the correct answer on every arrow and sequence task. The BOS-only edit recovers more on the sequence tasks and arrow_double; the copied-pattern edit recovers more on arrow_add, arrow_subtract, and addition, where the BOS-only edit fails. We define % recovery as $(p_{\text{edit}} - p_{\text{low}})/(p_{\text{high}} - p_{\text{low}})$.

The first intervention is the BOS-only edit. To perform it, we first run both prompts without intervention and record attention patterns. We then rerun the low-evidence prompt and edit the attention pattern of all heads, at every query position. Denote by $\alpha_{q\to k}$ the attention of a head in a low-evidence prompt from position q to k. We set each head's attention to the BOS token (originally $\alpha_{q\to 0}$) to the BOS-attention of the same head in the corresponding position (q') in the high-evidence prompt ($\alpha'_{q'\to 0}$). We then rescale all of the other attention entries ($k > 0$) as

$$\tilde{\alpha}_{q\to k} = \alpha_{q\to k}\frac{1-\alpha'_{q'\to 0}}{1-\alpha_{q\to 0}}$$

The edit thus changes only how much attention goes to BOS versus the other tokens; we do not touch the heads' values, which flow like normal. We leave a query unedited when $1-\alpha_{q\to 0} \le 10^{-3}$, because the scale factor is then unstable; this affects 0.5–0.8% of queries.

We also apply the *copied-pattern edit* that we used in the Number Game (Appendix L), to all heads at every query position. It sets the BOS attention in the same way, and also copies the other attention entries from the high-evidence prompt:

$$\tilde{\alpha}_{q\to k} = \alpha'_{q'\to k'}\frac{1-\alpha'_{q'\to 0}}{\sum_{j>0}\alpha'_{q'\to j'}},$$

for $k > 0$, where k' is the high-evidence position of the token at k; the sum runs over the tokens of the low-evidence prompt. Again, we do not edit a query when the high-evidence attention on these tokens is zero; this excludes only 0.5-0.8% of queries.

For both interventions, we measure performance as the probability that Llama assigns to the correct answer token at the final position. We record this probability, averaged over the 64 draws, for the low-evidence prompt (p_{low}), the high-evidence prompt (p_{high}), and the edited low-evidence prompt (p_{edit}); we report the last as % recovery: $\frac{p_{\text{edit}}-p_{\text{low}}}{p_{\text{high}}-p_{\text{low}}}$.

We find that Llama can do each task with increasing accuracy as it moves from the low evidence (few examples) to high evidence (many examples) setting (Table 2). The BOS-only intervention is very effective on ascending, descending, or geometric sequences, which are similar in formatting to the Number Game, although they involve a progression of numbers, rather than a set, which is unordered. This attention spreading also works on the arrow-formatted doubling task. It works much less effectively on the addition or subtraction arrow-formatted tasks, and fails entirely on addition.

By contrast, the copied pattern edit is more broadly effective, but with a lower effect size. % recovery ranges from 25-75%, with strong effects on the last two sequence tasks, but weak effects on the first. On the addition task, which previously saw no recovery, we now see 49% recovery. In summary, both of these interventions have some effects, but the size of this effect, and which tasks are affected,

varies. Given the effectiveness of these interventions, we find it likely that the spreading of attention also influences model output on these tasks, although the mechanism is clearly not identical.

N THE FULL NEW MODEL

In this section we provide an explanation of the new model that we propose. We first provide an account of how this model can be understood in Bayesian terms. Then we explain the technical details regarding the training of this model.

N.1 BAYESIAN INTERPRETATION

Let X be an input, and $y \in \{1, \ldots, 999\}$, be the next number that Llama is predicting. We split the hypothesis space H into hypothesis families $f \in F$, one per belief manifold, and write H_f to denote the hypotheses of family f. Each family marginalizes over its own hypotheses.

We can formulate Llama's behavior as a sort of message-passing algorithm. The predictions made by each family (manifold) combine by log-linear pooling of per-family posterior predictives: their log-messages add in the residual stream, as a product of experts. We write this as follows:

$$\ell(y \mid X) = \sum_{f \in F} \log m_f(y \mid X), \tag{11}$$

$$\log m_f(y \mid X) = \log \left[\sum_{h \in H_f} p(h \mid X, f)\, p(y \mid h, f) \right], \tag{12}$$

$$p(y \mid X) = \mathrm{softmax}_y\, \ell(y \mid X). \tag{13}$$

Here, $m_f(y|X)$ denotes the message from family f. Messages need not be normalized; their log-values are understood up to an additive constant independent of y.

How could this be computed? We next outline how Llama could actually implement the message-passing algorithm described above, computing $p(h|X, f)$ and $p(y|h, f)$.

1. Belief manifold construction heads average the numbers' features into a position on the manifold. Let $\phi_f(x)$ be the features of number x on the manifold of family f. At the final space token, belief manifold construction heads average them:

$$\Phi_f(X) = \frac{1}{|X|} \sum_{x \in X} \phi_f(x).$$

Evidence accumulation heads move attention from BOS to non-BOS tokens as $|X|$ grows. This means that the norm of its position on the manifold scales with

$$A_f(|X|) = \frac{|X|}{|X| + \tau_f}.$$

For families besides intervals, $\phi_f(x)$ has a 1 for each hypothesis that contains x and 0 elsewhere, so the match between the input and hypothesis h,

$$\Pi_h = \langle A_f(|X|)\, \Phi_f(X),\ \phi_f(h) \rangle = A_f(|X|)\, f_h,$$

is the fraction f_h of X in h, scaled by the evidence.

2. Hypothesis writer MLPs compute an unnormalized posterior over the family's hypotheses. We define H_f to contain both the family's rules and a background hypothesis $\varnothing$, a uniform prior on all numbers $1, \ldots, 999$. If we learn a log-likelihood weight γ_h and a log-prior β_h per hypothesis, we can formulate $\log \widetilde{p}(h|X, f)$ as:

$$\log \widetilde{p}(h \mid X, f) = \gamma_h\, \Pi_h + \beta_h, \qquad \log \widetilde{p}(\varnothing \mid X, f) = \beta_\varnothing - \gamma_\varnothing\, A_f(|X|) \log(999).$$

The background's score $\log \tilde{p}(\varnothing|X, f)$ is the log-likelihood of X under uniform numbers, scaled by the same evidence. The likelihood for actual hypotheses $\log \tilde{p}(h|X, f)$ is noise-tolerant: a number outside h lowers h's score rather than eliminating h. This contrasts with the Bayesian model's indicator function $\mathbf{1}[X \subseteq h]$; however, it is closer to the likelihood formulated by Goodman et al. (2008), which defines the likelihood as proportional to e^{-bQ}, where Q is the number of examples that violate the rule, and b (the outlier penalty parameter) is analogous to γ_h.

3. The hypothesis writer MLPs marginalize over the hypotheses to write a message. We model a hypothesis h as assigning uniform probability to all of its members, and leaking a fraction ϵ of its mass to the prior over numbers, which we take to be a uniform prior over all numbers. This yields the following posterior predictive:

$$p(y \mid h, f) = (1-\epsilon)\frac{\mathbb{1}[y \in h]}{|h|} + \epsilon\, p(y), \qquad p(y \mid \varnothing, f) = \frac{1}{999}.$$

We suggest that an MLP computes the per-family message by marginalizing over the hypotheses in H_f:

$$\log m_f(y \mid X) = \log \left[\sum_{h \in H_f} \widetilde{p}(h \mid X, f)\, p(y \mid h, f) \right].$$

This is difficult to compute, because it contains $\widetilde{p}(h \mid X, f)$, a probability, rather than a log probability. However, we do not claim that an MLP computes this probability explicitly. Rather, consider the MLP's residual write; this is $\Delta r_f = \sum_j a_j(X)\, d_j$ over neurons j. Passing this through the unembedding U, its direct logit write is

$$\ell_{\mathrm{MLP}\,f}(y \mid X) = (U \Delta r_f)_y = \sum_j a_j(X)\ \mathrm{logit}_j(y), \qquad \mathrm{logit}_j(y) = (U d_j)_y.$$

We hypothesize that this write *approximates* the family's log-message:

$$\sum_j a_j(X)\ \mathrm{logit}_j(y)\ \approx\ \log \left[\sum_{h \in H_f} \widetilde{p}(h \mid X, f)\, p(y \mid h, f) \right].$$

Note that the neurons' activations a_j may depend on more than just the model's current belief as represented on the manifold; the context consistency circuit, for example, may feed them too. Thus, there is considerable flexibility in how the MLP approximates this.

As long as the background's score exceeds that of every hypothesis the message is nearly flat. When a hypothesis' score passes the background's, its weight grows exponentially in $\gamma_h \Pi_h$ until it holds most of the posterior.

4. The MLP writes add in the residual stream. Finally, the writes of the MLPs for each family combine in the residual stream, before the final RMSNorm and unembedding:

$$\ell(y \mid X) = \sum_{f \in F} \log m_f(y \mid X); \quad p(y \mid X) = \mathrm{softmax}_y\, \ell(y \mid X)$$

Note that each family here need not have its own MLP. MLPs are likely computing the messages of multiple families in parallel using different neurons, and these messages will still sum together in the residual stream in the same way.

Thus, the averaging attention heads compute the input X's position on the manifold, while the MLPs compute unnormalized posteriors over the family's hypothesis, marginalize over hypotheses, and finally write to the residual stream.

N.2 Experimental Details of the New Model

Here, we detail how our new model is formulated and learned.

Model. Our model replaces the Bayesian model's single hypothesis space $\mathcal{H}$ with several *families* of hypotheses, $f \in F$. Each family is a small Bayesian model with its own hypotheses $\mathcal{H}_f$, its own posterior, and its own posterior predictive $p_f(y|X)$. The families' predictions are multiplied:

$$p(y \mid X) = \operatorname*{softmax}_y \sum_{f \in F} \log p_f(y \mid X), \qquad p_f(y \mid X) = \sum_{h \in \mathcal{H}_f} p(h \mid X, f)\, p(y \mid h). \tag{14}$$

All families share an evidence accumulation curve, $A(N) = \tau N/(N+\tau)$, where τ is learned.

Within-family posteriors. For each hypothesis h besides magnitude and the prior (see families below), we compute the fraction $f_h(X)$ of X that falls in h, including repeats:

$$\log p(h|X, f) = \gamma_h\, A(|X|)\, f_h(X) + \beta_h + \text{const}, \tag{15}$$

where const is the appropriate normalizing constant. Every family also contains a background hypothesis $\varnothing$, a uniform prior over all numbers, which lets a family switch itself off. It scores X's log-likelihood under uniform numbers, scaled by the same evidence scaling curve A. That is, $\log p(\varnothing \mid X, f) = \beta_{\varnothing,f} - \gamma_{\varnothing,f}\, A(|X|) \log(999)$. This hypothesis predicts $p(y \mid \varnothing) = 1/999$. Each family has its own $\beta_{\varnothing,f}$ and $\gamma_{\varnothing,f}$. In this formulation, if X contains a number x not in h, this only lowers h's score rather than eliminating it, as in the Bayesian model; once f_h is too low, the background hypothesis wins out.

Predictions. Each set hypothesis predicts its members uniformly, following the size principle, and gives a fraction ϵ of its mass to a small-number prior $p_0(y) \propto y^{-\lambda}$, akin to that of the Bayesian model. In the following formulation, ϵ and λ are learned and shared by all families:

$$p(y \mid h) = (1-\epsilon)\, \frac{\mathbb{1}[y \in h]}{|h|} + \epsilon\, p_0(y). \tag{16}$$

Families. Our model contains the following six families (Table 3).

Clock. This family contains 17 hypotheses, i.e., each residual for $x = n \mod 2$, $x = n \mod 5$, and $x = n \mod 10$. Each number x falls in three of the hypotheses. Remainders 1-4 mod 5 share one γ and one β, as do remainders 1-4 and 6-9 mod 10 jointly. The other remainders have their own γ and β.

Powers of 2. This family contains 4 hypotheses: multiples of 4, 8 and 16, as well as the numbers that are not multiples of 4.

Multiples of 3. This family contains 4 hypotheses: multiples of 3 and 9, and the numbers that are not multiples of 3.

Starts-with-n. This family contains the 9 *starts-with-n* hypotheses, scored by the fraction of the list that starts with n. A hypothesis first picks a digit length t for y, then a t-digit number starting with d uniformly: $p(y \mid d) = \theta_{t(y)}(X)\, \mathbb{1}[y \text{ starts with } d]/n_{d,t(y)}$, where $n_{d,t}$ is the number of t-digit numbers starting with d. $\theta_t(X) = (N q_t(X) + a\, c_t)/(N+a)$ is the list's fraction q_t of t-digit numbers, smoothed toward $c = (1, 10, 100)/111$ with a learned strength a.

Magnitude. This family captures lists whose numbers lie in one region of 1-999. Its one hypothesis (besides $\varnothing$) is that the numbers come from a Gaussian with an unknown center and width. To keep the Gaussian on a fixed scale, we divide every number by 999, so that it lies between 0 and 1; we write $u = x/999$.

We place a prior over the mean μ and the standard deviation σ: a normal-inverse-gamma prior, with learned parameters $\mu_0, \kappa_0, \alpha_0, \beta_0$. We update it based on X's mean $\bar{u}$ and variance S, and counts X as having $n = A(|X|)$ examples:

$$\kappa_n = \kappa_0 + n, \quad \mu_n = \frac{\kappa_0 \mu_0 + n\bar{u}}{\kappa_n}, \quad \alpha_n = \alpha_0 + \frac{n}{2}, \quad \beta_n = \beta_0 + \frac{nS}{2} + \frac{\kappa_0 n(\bar{u} - \mu_0)^2}{2\kappa_n}. \tag{17}$$

The hypothesis's score measures how well a Gaussian explains the input, averaged over all centers and widths the prior allows (the marginal likelihood), multiplied by a learned γ:

$$s(\text{mag} \mid X) = \gamma \Big[\log \frac{\Gamma(\alpha_n)}{\Gamma(\alpha_0)} + \alpha_0 \log \beta_0 - \alpha_n \log \beta_n + \tfrac{1}{2} \log \frac{\kappa_0}{\kappa_n} - \tfrac{n}{2} \log 2\pi - n \log 999 \Big]. \tag{18}$$

A tight cluster of numbers scores high; numbers spread over 1-999 score low. The last term, $-n \log 999$, makes this score comparable to the background's. Note that Gaussian is continuous and gives a probability per unit of u, not per number. Neighboring numbers are $1/999$ apart in u, so each number gets about $1/999$ of that value, which is what the term subtracts.

To predict, the hypothesis uses a Student-t with $2\alpha_n$ degrees of freedom, center μ_n, and squared scale $\beta_n(\kappa_n + 1)/(\alpha_n \kappa_n)$), evaluated at each $y = 1, \ldots, 999$ and normalized to sum to 1.

Prior. The prior family contains only one hypothesis, i.e. that the numbers are drawn from the background distribution, p_0. This hypothesis is scored by X's average log-likelihood under p_0, $\gamma\, A(N)\, \frac{1}{N} \sum_{x \in X} \log p_0(x)$. The hypothesis predicts $p_0(y)$. With few examples its prior dominates, and the family predicts small numbers. With more examples, inputs spread over all numbers favor the background and turn the family off, while lists of small numbers keep it on.

family	hypotheses besides $\varnothing$	γ	β
clock	last digit 0–9; remainder 0–4 mod 5; even; odd	14	6
powers of 2	multiples of 4, 8, 16; the rest	4	3
multiples of 3	multiples of 3, 9; the rest	3	2
leading digit	starts with 1–9	9	8
magnitude	Gaussian on $x/999$ (normal-inverse-gamma prior)	1	–
prior	numbers drawn from p_0	1	–
Total		32	19

Table 3: Count of parameters γ, β per family. Note that one prior β per family is fixed at 0 as a reference prior.

Parameter Count. The model has 71 free parameters: 32 scale parameters γ_h, 19 priors β_h, 12 background parameters ($\beta_{\varnothing,f}, \gamma_{\varnothing,f}$ for each family), the evidence scale τ, the four normal-inverse-gamma parameters, ϵ, λ, and the length smoothing a.

Fitting the Model. We fit on the same data and loss as the Bayesian model (Equation (7)): the population-level forward KL, averaged over the 39 concepts and $N = 1, \ldots, 30$, on the 100 training lists per cell. Because fits of this model can be unstable, we run 32 fits. The first starts from fixed initial values; the others perturb the initial values of the per-hypothesis γ_h and β_h. Each start runs 200 full-batch Adam steps (learning rate 0.05, reduced by 70%, at 60% of the steps and 85% of the steps), followed by 300 L-BFGS iterations (Liu & Nocedal, 1989). We continue training only the 8 starts with the lowest training loss, up to 1500 L-BFGS iterations; we discard the others. We report the mean over these 8 fits on the held-out data, and use the fit with the lowest training loss for figures.

O LLAMA-3.1 70B

We also re-perform our representational geometry experiments on Llama 3.1 (70B), to show that our results also hold for a larger model. Because all of the methods are the same, we do not repeat them here; please see the main text and other appendices for these.

Behaviorally, Llama 70B behaves much like 8B, successfully playing the Number Game (Figure 36, left). We note that it is more capable of learning some of the rules that 8B struggled with, such as *multiples-of-14* or *7* (Figure 36, right). This suggests that larger-capacity models may indeed learn more difficult hypotheses, though we do not investigate this at a representational level. We note that the individual outputs from Llama 70B look no less noisy and no better formed than those from Llama 8B (Figure 37).

On the manifold side, we once more use DAS to find where the manifolds reside. Because DAS on Llama 70B is much more expensive than on 8B—the former has over double the number of layers, and is in general much slower to run—we instead run the k-sweep on only one layer, layer 56/72, and then choose k in that way. This preliminary test (not shown) suggested that Llama 70B benefits from a slightly larger $k = 48$, though $k = 32$ would also be a reasonable choice. Based on this, we

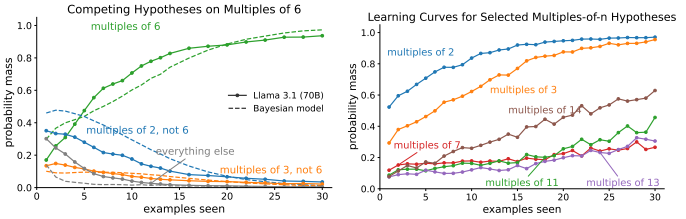


Figure 36: Llama-3.1 (70B)'s behavior on *multiples-of-6* input (left) and various other inputs (right). Much like Llama-3.1 (8B), its *multiples-of-6* is reasonably well fit by a Bayesian model, though somewhat less so.

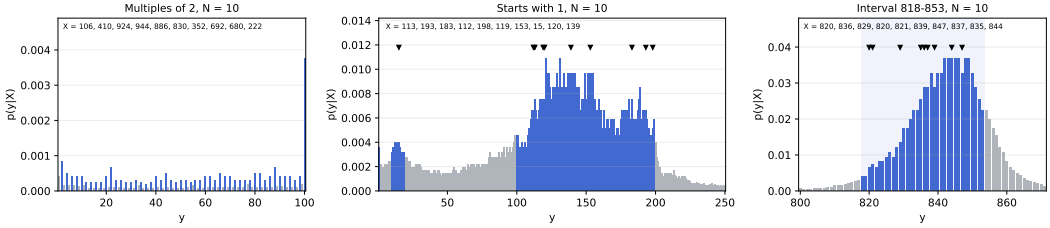


Figure 37: Instance-level outputs from Llama-3.1 (70B), on *multiples-of-2* (left), *starts-with-1* (center), and the interval [818,853] (right).

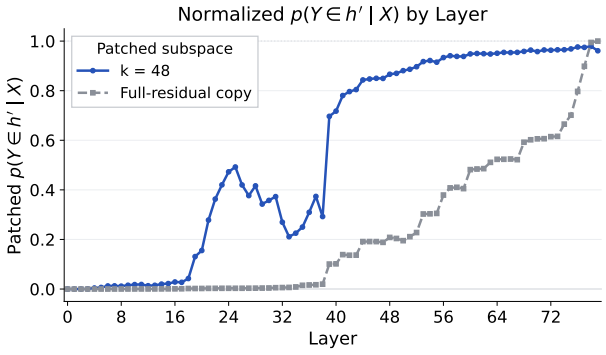


Figure 38: DAS performance vs. layer, for a 48D DAS subspace on Llama-3.1 (70B).

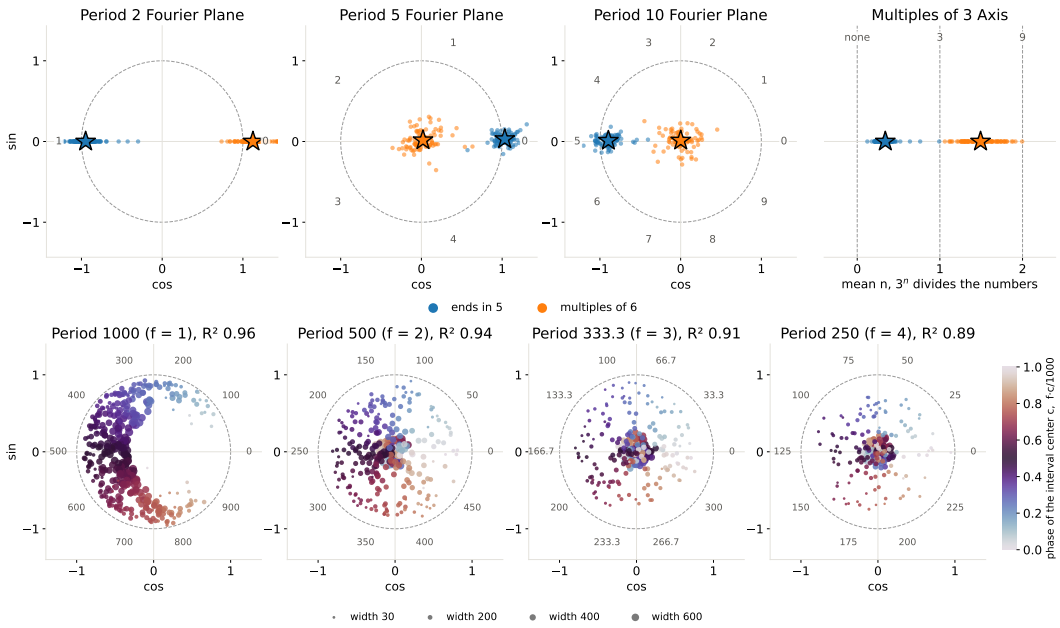


Figure 39: **Top**: *multiples-of-n* and *ends-in-n* manifolds in Llama-3.1 (70B). **Bottom**: Interval representations in Llama-3.1 (70B).

run a full DAS sweep across layers, observing a small peak in DAS effectiveness around layer 25, and a larger, sustained jump at layer 40 that only gradually increases to full performance (Figure 38).

On the basis of this, we search for manifolds at a relatively late layer, layer 66, recognizing that an earlier layer might also work. Our results (Figure 39) suggest that Llama (70B) similarly has Fourier clocks for *multiples-of-n* and *ends-in-n*, while recording interval centers using longer-period Fourier features. We note that compared to 8B, 70B better captures the irregular periods like $5/2$ and $10/3$, though this may simply be a result of using a larger k.

We then perform steering using (models of) these manifolds as before. Our results (Figure 40) suggest that these manifolds are also causally active: steering works just as well on Llama 70B as it does on 8B.

We also show that Llama 70B represents ambiguity similarly to Llama 8B, by plotting its representations of mixed / ambiguous inputs. The results (Figure 41) are similar to those of Llama 8B, with the representations walking the chord between the points representing purely 4 or 7 mod 10. However, we note that there is bowing away from the Fourier plane's center in the 3rd harmonic. We have no current explanation for this behavior, though understanding this sort of phenomenon would be an interesting goal for future work.

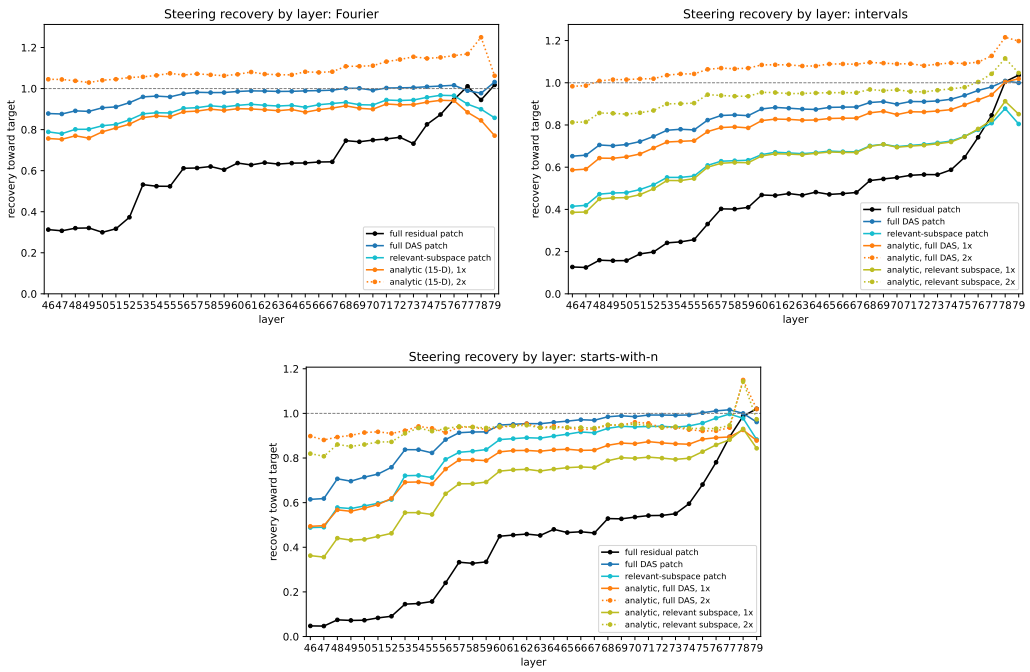


Figure 40: Steering results for *multiples-of-n* and *ends-in-n* (top left), intervals (top right), and *starts-with-n* (bottom) for Llama-3.1 (70B).

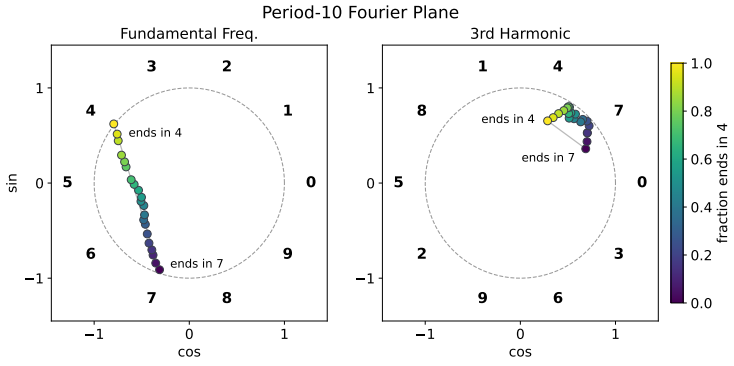


Figure 41: Llama-3.1 (70B)'s representation of a mixed *ends-in-4/7* input on the period-10 Fourier clock, plotted with respect to its fundamental frequency (left) and its 3rd harmonic (right).