Extended Abstract Track

Constrained Belief Updating and Geometric Structures in Transformer Representations

Editors: List of editors' names

Abstract

How do transformers trained on next-token prediction represent their inputs? Our analysis reveals that in simple settings, transformers form intermediate representations with fractal structures distinct from, yet closely related to, the geometry of belief states of an optimal predictor. We find the algorithmic process by which these representations form and connect this mechanism to constrained belief updating equations, offering insight into the geometric meaning of these fractals. These findings bridge the gap between the model-agnostic theory of belief state geometry and the specific architectural constraints of transformers.

Keywords: computational mechanics, mechanistic interpretability, belief state geometry

1. Introduction

What representations should we expect in transformers, and how do they relate to training data structure? Recent work by Shai et al. (2024) shows that transformers linearly represent belief state geometries—the geometry associated with Bayesian updating of beliefs over the hidden states of the data-generating process. This model-agnostic theory doesn't specify how transformers construct this geometry.

To study how transformers do this, we analyze networks trained on a class of data-generating processes and find highly structured patterns in the intermediate activations of these transformers. We then perform mechanistic interpretability on the attention heads of these transformers to find a common algorithm that builds the geometry of intermediate representations. Finally, we propose a theoretical explanation for why these particular geometries occur in the intermediate activations in the transformer, by taking the model-agnostic belief state geometry equations and constraining them by the form of the attention mechanism[1].

2. Methodology

In order to study how transformers build up belief state geometries, we investigate the Mess3 class of Hidden Markov Models (HMMs) (Marzen and Crutchfield (2017)). HMMs in this class, depicted in Figure 1(a), have three hidden states and are parametrized by two parameters, α and x, which control the emission and hidden transition probabilities respectively. Shai et al. (2024) have shown that transformers linearly represent the belief state geometry when trained on data generated from these HMMs. The belief state geometry is given by the probabilities (plotted in a probability simplex) over the hidden states of the HMM that a Bayesian observer would have upon seeing strings of data generated by the HMM. An example is shown in Figure 1(b).

1. Code available at https://github.com/dummy/neurreps.

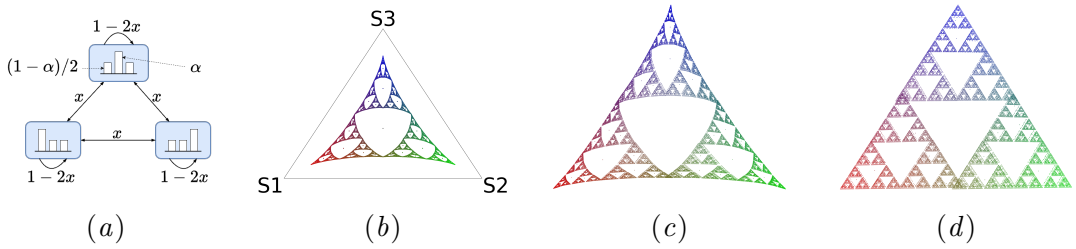


Figure 1: The model's internal representations exhibit complex geometric structure matching the belief state geometry. (a) The Mess3 HMM, vertices represent hidden states with their emission distributions. (b) The ground truth belief state geometry of Mess3. Each point represents a belief state over the hidden states of the HMM of an optimal observer of emissions, with distances to the vertices of the simplex corresponding to the probabilities of the three hidden states. (c) The PCA projection of the model's residual stream before unembedding reveals a geometric representation that closely matches the belief geometry. (d) The PCA projection of the intermediate residual stream after attention exhibits an intricate but different, structure. In (b-d), points are colored according to the ground-truth belief states, taking the 3 probabilities as RGB values.

In each experimental run, we specify an (α, x) pair and generate sequences from the HMM. We then train a standard transformer (see Appendix B for details) on these sequences. We then apply Principal Component Analysis (PCA) to the residual stream activations across all possible inputs and perform mechanistic interpretability analysis. The activations are well-captured by a few principal components (see Appendix D), enabling low-dimensional visualization and analysis.

3. Results

3.1. Intermediate representations are fractals, but not belief state geometry

We examine the intermediate representations in the residual stream (after the attention but before the MLP) of the transformers as we vary the parameters of the data-generating HMM. We find that these representations were fractals, but differed from those observed in the final residual stream after the MLP (Figures 1 and 3). While the final representations align with the belief state geometry, the intermediate fractals are distinct. The following results explain how these intermediate representations are constructed and provide a theoretical explanation for their unexpected structure.

3.2. Intermediate representations are built by algorithms in the belief simplex

We find that attention performs an algorithm with a direct interpretation in the belief simplex. Figure 2 illustrates the computation of the current location within the belief geometry based on the observation of past tokens.

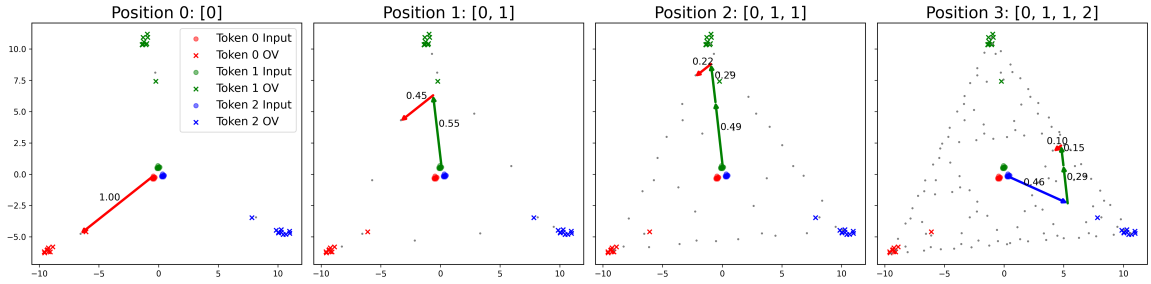


Figure 2: Intermediate representation construction. From left to right, we show how attention constructs the intermediate representations for 4 example input-sequences of increasing length. The token embeddings lie near the origin, and the OV-projections lie towards the corners of a triangle. Treating these as vectors, attention works by taking linear combinations of these three vectors in order to build up the fractal. Vectors show the components of the sum for each example, while gray dots show all possible vector-sums for all possible sequences in that position.

The attention operation in transformers consists of two circuits: the query-key (QK) circuit, which determines what parts of the input sequence to attend to, and the output-value (OV) circuit, which specifies what information should be read from the attended tokens (Elhage et al. (2021)).

Projecting token embeddings onto PCA space reveals three clusters that lie close to the origin. Meanwhile, OV-values lie in three clusters whose directions from the origin form the vertices of a triangle[2], naturally interpreted as the vertices of the belief simplex. The OV circuit projects token embeddings symmetrically around the simplex vertices, forming update vectors (Figure 2).

The model updates its current position by adding these vectors, with magnitude determined by the QK attention weight, into the current token position. The attention pattern is invariant to token value, decaying with distance to the current token. The attention module independently integrates information from a finite number of past tokens to compute the current location within the simplex. As the attention weight decays with distance, the impact of past tokens on the current belief state diminishes over time.

3.3. Relating Intermediate Representations to Belief Updating Equations

The interpretation of attention as operating in the belief simplex suggests a connection to the theory of belief updating. Since the OV circuit is only able to access information from the source token that is attended to, we can write a constrained belief updating equation that sums contributions from the value of the token n places back for each value of n, assuming the initial belief is the stationary distribution of the HMM, η_∞. This gives the

2. Their direction from the origin is independent of sequence position, and their distance from the origin are also mostly independent of sequence position, but note that the OV-embeddings for the first position are closer to the origin, as show in (Figure 2), left. We believe this would not occur if we used a BOS-token.

following equation for the constrained belief at position L in the sequence:

$$r^{(L)} = \eta_\infty + \sum_{n=0}^{L-1} \left(\eta_\infty T^{|a_{L-n}} T^n - \eta_\infty \right) \tag{1}$$

where T is the HMM's hidden state transition matrix, and $T^{|a_i}$ is the HMM transition matrix conditioned on seeing token a in the i-th position (see Appendix A for details).

This equation generated the ground truth intermediate representations in Figure 3. The formula accurately predicts the intermediate structure for $\alpha \in [0.2, 0.6]$. For α outside this range, the predictions deviate from the observed representations consistently. Further investigation is needed to fully characterize the model's behavior across all α values.

The attention pattern relates to powers of the Markov transition matrix of the hidden process, T^n, where n is the token distance. See Appendix C for a detailed analysis of the eigenvalues and their implications for the model's behavior.

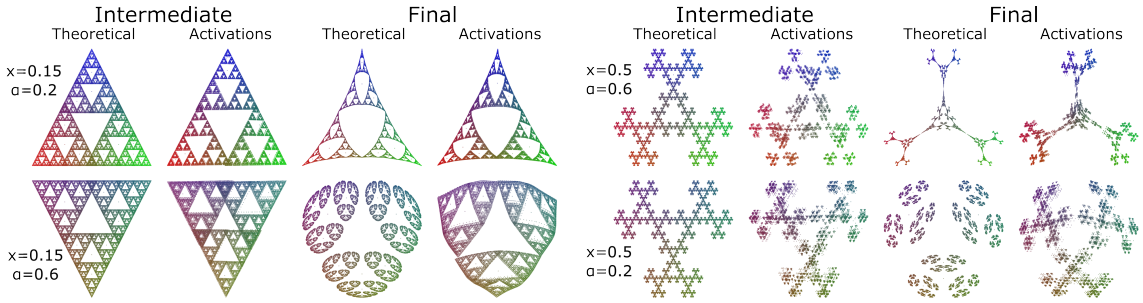


Figure 3: Comparison of model representations and theoretical predictions for different Mess3 hyperparameters. Each subfigure shows four columns: 1. Intermediate representation from (1). 2. Projection of the model activations in the intermediate layer. 3. Ground truth belief states. 4. Projection of the final representation.

4. Conclusion

In this work, we have explored the connection between the belief geometry of an optimal predictor for a simple stochastic process and the internal representations learned by a transformer model trained on sequences generated from this process. Our analysis reveals that the model's intermediate representations closely resemble the (constrained) belief states predicted by our theory, despite the inherent limitations of the parallel transformer architecture in directly implementing recursive computations.

The consistency and predictability of the intermediate geometry across different model realizations suggest that there may be a deeper theoretical explanation for the model's behavior. While we have observed strong connections between the model's representations and the optimal predictor's belief states, a complete theoretical understanding of how the model arrives at these representations remains open.

References

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. https://transformer-circuits.pub/2021/framework/index.html.

Sarah E. Marzen and James P. Crutchfield. Nearly maximally predictive features and their dimensions. *Physical Review E*, 95(5), May 2017. ISSN 2470-0053. doi: 10.1103/physreve.95.051301. URL http://dx.doi.org/10.1103/PhysRevE.95.051301.

Adam S. Shai, Sarah E. Marzen, Lucas Teixeira, Alexander Gietelink Oldenziel, and Paul M. Riechers. Transformers represent belief state geometry in their residual stream, 2024. URL https://arxiv.org/abs/2405.15943.

Appendix A. Mathematical Details of HMMs and Belief State Geometry

In this work we created training data from a class of Hidden Markov Models (HMMs) called Mess3. The HMMs have three hidden states $\mathcal{S} = \{1, 2, 3\}$ and emit from a vocabularly of three tokens $\mathcal{X} = \{0, 1, 2\}$.

The HMMs in this class are parameterized by α and x, with dependent quantities $\beta = (1 - \alpha)/2$ and $y = 1 - 2x$.

The labeled transition matrices define the probability of moving to state j (indexing columns) and emitting the token on the label, a, conditioned on being in state i (indexing rows), $P(s_j, a|s_i)$ and are:

$$T^{(0)} = \begin{bmatrix} \alpha y & \beta x & \beta x \\ \alpha x & \beta y & \beta x \\ \alpha x & \beta x & \beta y \end{bmatrix} \tag{2}$$

$$T^{(1)} = \begin{bmatrix} \beta y & \alpha x & \beta x \\ \beta x & \alpha y & \beta x \\ \beta x & \alpha x & \beta y \end{bmatrix} \tag{3}$$

$$T^{(2)} = \begin{bmatrix} \beta y & \beta x & \alpha x \\ \beta x & \beta y & \alpha x \\ \beta x & \beta x & \alpha y \end{bmatrix} \tag{4}$$

Note that even though the dynamics amongst the emmissions are infinite-Markov order, the dynamics amongst the hidden states are Markov, with a transition matrix given by marginalizing out the token emissions: $T = \sum_{a \in \mathcal{X}} T^{(a)}$.

We can also define a conditional transition matrix, $T^{|a}$, with elements $T^{|a}_{i,j} = P(s_j|a, s_i)$, which is given by normalizing each labeled transition matrix such that every row sums to 1.

An important part of the work presented here is about how an optimal observer of token emissions from the HMM would update their beliefs over which of the hidden states the HMM is in, given a token sequence. If the observer is in a belief state given by a probability distribution η (a row vector) over the hidden states of the data-generating process, then the update rule for the new belief state η' given that the observer sees a new token a is:

$$\eta' = \frac{\eta T^{(a)}}{\eta T^{(a)} \mathbf{1}} \tag{5}$$

where $\mathbf{1}$ is a column vector of ones of appropriate dimension, with the denominator ensuring proper normalization of the updated belief state. In general, starting from the initial belief state η_∞, we can find the belief state after observing a sequence of tokens $a_0, a_1, \ldots, a_N$:

$$\eta = \frac{\eta_\infty T^{(a_0)} T^{(a_1)} \cdots T^{(a_N)}}{\eta_\infty T^{(a_0)} T^{(a_1)} \cdots T^{(a_N)} \mathbf{1}} \ . \tag{6}$$

For stationary processes, the optimal initial belief state is given by the stationary distribution η_∞ over hidden states of the HMM (the left-eigenvector of the transition matrix $T = \sum_x T^{(a)}$ associated with the eigenvalue of 1).

The beliefs have a geometry associated with them, called the belief state geometry. The belief state geometry is given by plotting the belief distributions over all possible sequences of tokens generated by the HMM in the probability simplex.

Also note that the constrained belief updating equation given in Eq. (1) is a natural geometric representation of $\Pr(S_{L+1}) + \sum_{n=0}^{L-1} \big[\Pr(S_{L+1}|X_{L-n} = a_{L-n}) - \Pr(S_{L+1})\big]$.

Appendix B. Model architecture and training procedure

We employ a standard single-layer transformer model with learned positional embeddings. The model architecture follows the conventional transformer design [Citation], with $d_{\text{model}} = 64$ and $d_{\text{ff}} = 256$. Depending on the Mess3 parameters, we use either a single-head or a double-head attention mechanism. We conduct a systematic sweep over the HMM parameters α and x, training a separate model for each pair. Models are trained on next-token prediction using cross-entropy loss, with batch size 128. We use Adam optimizer with a 10^{-4} learning rate and no weight decay. Each model is trained for approximately 15 million tokens.

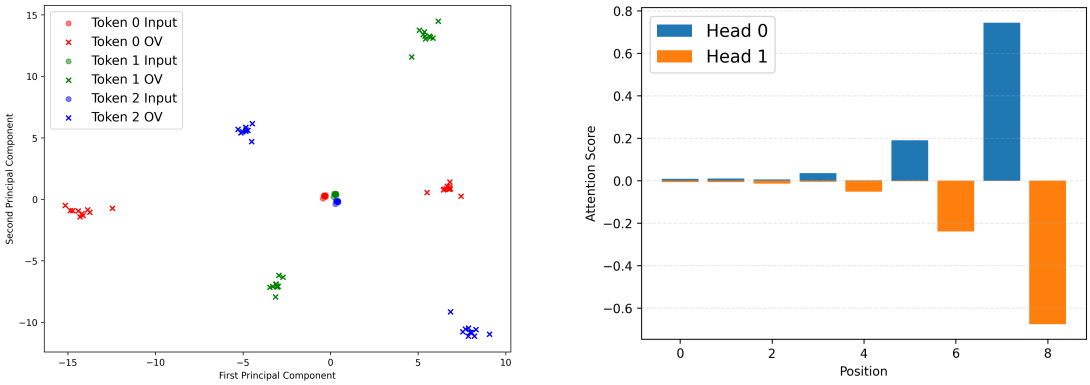
We generate all possible input sequences up to length 10, recording hidden activations from the transformer's residual stream. These activations are organized into a dataset capturing the model's response to all input patterns.

Input sequences consist of three symbols, embedded with positional information, without a beginning-of-sequence (BOS) token.

Appendix C. Implications of Transition Matrix Eigenvalues

The attention pattern in our model relates to powers of the Markov transition matrix of the underlying hidden states (marginalizing out the emissions), T^n, where n is the token distance. The matrix T has eigenvalues $\lambda_1 = 1$ and $\lambda_2 = \lambda_3 = 1 - 3x$. We observed that the attention weight n tokens back is approximately $\lambda_3^n = (1 - 3x)^n$. For the transition matrix

T to be row stochastic (a requirement for a valid HMM), x must be in the range $[0, 0.5]$. Interestingly, when $\lambda_3 < 0$ (which occurs when $x > 1/3$), the predicted pattern oscillates and cannot be captured by a single attention head, since attention pattern entries must be non-negative. In these cases, we observe that a single-head transformer captures an incomplete representation of the belief state geometry. However, upon adding a second attention head, the model converges to the solution predicted by the belief updating equation, even in the presence of oscillatory dynamics, as shown in Figure 4. Thus, our analysis here gives us a handle to relate the architectural constraints of the attention mechanism to the structure of the training data.



(*a*) Token embedding space

(*b*) Attention patterns of the two heads

Figure 4: Attention heads combine to capture oscillatory dynamics in belief updating. (a) In the token embedding space, the model uses each attention head to embed tokens on opposite poles of the simplex. (b) The attention patterns of the two heads act as positive and negative components. When combined, they produce the oscillatory pattern predicted by $\lambda_3^n = (1 - 3x)^n$.

Appendix D. Dimensionality of Residual Stream Activations

We perform PCA on the residual stream activations after the attention module (intermediate) and before the unembedding layer (final). The effective dimensionality of the residual stream is low, with the first few components capturing most of the variance (Table 1). In most cases, the first 3 components explain over 90% of the variance. For $x = 0.5$, the effective dimensionality is higher, possibly due to the oscillatory dynamics of the belief updating equation in this regime. Further investigation is needed to fully understand this phenomenon.

Figure 3 depicts the belief geometry using the first 3 principal components. To enable consistent comparison of the learned representations across different model configurations, we perform a regression to find a projection that aligns the principal components with the ground truth belief geometry.

Table 1: Cumulative explained variance ratios for PCA components of the residual stream activations at the intermediate position (after attention) and the final position (before unembedding). The table shows results for different settings of the Mess3 HMM parameters x and α.

		Intermediate				Final			
	α	0.15	0.15	0.5	0.5	0.15	0.15	0.5	0.5
component	x	0.2	0.6	0.6	0.2	0.2	0.6	0.6	0.2
0		0.5408	0.4648	0.4074	0.5268	0.9618	0.4947	0.4596	0.6503
1		0.8768	0.8894	0.8028	0.8519	0.9825	0.7681	0.7096	0.8592
2		0.9673	0.9859	0.8913	0.9173	0.9943	0.9811	0.8855	0.9689
3		0.9749	0.9903	0.9455	0.9649	0.9960	0.9897	0.9189	0.9755
4		0.9815	0.9929	0.9848	0.9886	0.9969	0.9916	0.9428	0.9807
5		0.9870	0.9942	0.9978	0.9977	0.9976	0.9931	0.9586	0.9850
6		0.9914	0.9955	0.9986	0.9984	0.9981	0.9945	0.9723	0.9886