Addressed State Memory: A Mechanistic Analysis of Intervenable Attention

Justin Brown
Independent Researcher
digitaldaimyo@gmail.com

January 30, 2026

Abstract

Transformer attention mechanisms unify retrieval and selection into a single softmax operation, obscuring the internal structure of memory use. We introduce *Addressed State Memory* (ASM), an architecture that explicitly separates conservative memory construction from selective refinement through a two-stage access mechanism. Through geometric analysis and causal intervention on models up to 240M parameters, we show that refinement functions as directional suppression rather than additive reasoning. This behavior emerges early in training and remains stable across scales. ASM achieves competitive language modeling performance while enabling mechanistic clarity through its intervenable design.

Code and checkpoints: `https://github.com/digitaldaimyo/ASA`

Note: This is version 1. Matched-parameter baseline results and additional figures forthcoming in v2.

1 Introduction

Transformer attention mechanisms unify retrieval and selection into a single softmax operation over token representations. While this design has proven remarkably effective, it also obscures the internal structure of memory use: all context retrieval, competition, and suppression occur simultaneously and implicitly. As a result, it is difficult to disentangle how models store information over time, how they select among competing memories, and how these processes interact during training and inference.

These limitations become more pronounced in long-context and memory-augmented settings. Models that extend attention through recurrence, key–value caching, or external memory often struggle with credit assignment across time. In such systems, memory updates are either opaque (as in recurrent hidden states), brittle (as in unbounded caches), or heavily gated without explicit interpretability. While many architectures introduce mechanisms for retaining information, few provide explicit control over how retrieved memory is refined or suppressed once accessed.

In this work, we introduce *Addressed State Memory* (ASM), a memory-centric architecture that explicitly separates memory construction from memory utilization. ASM is built from *Addressed State Attention* (ASA) layers, which employ a persistent, slot-based memory that aggregates past information conservatively and monotonically. Access to this memory is performed in two stages: a first-order retrieval that produces a convex summary of stored state, followed by a second-order refinement operation that selectively reshapes this summary.

This separation enables a key property absent from standard attention: *intervenable refinement*. By isolating refinement as a distinct operation acting on retrieved memory rather than on tokens themselves, ASA allows direct mechanistic analysis and causal intervention. As we show, this structure reveals that refinement plays a suppressive, stability-enforcing role rather than acting as an additive reasoning mechanism.

Contributions. We make three primary contributions:

- We introduce Addressed State Attention (ASA), a memory-based attention mechanism with persistent slots and explicit two-stage access, and Addressed State Memory (ASM), an architecture built by stacking ASA layers.
- We demonstrate that ASM maintains competitive performance despite architectural constraints imposed for mechanistic clarity.
- We provide a mechanistic and causal analysis of ASA's refinement operation, showing that it functions as a stability-constrained suppressive correction rather than a source of new features.

Our results suggest that architectures designed for mechanistic clarity can both improve performance and enable deeper understanding of learned computation.

2 Addressed State Attention

Addressed State Attention (ASA) replaces token-to-token self-attention with interaction between tokens and a fixed set of persistent memory slots. Each ASA layer maintains K slot states per head, which summarize past context and are reused across sequence positions. Unlike recurrent or cache-based mechanisms, ASA does not overwrite memory nor store past tokens verbatim; instead, it incrementally constructs each slot as a causal, numerically stable aggregation of routed token representations.

2.1 Write Operation

At each timestep t, the input representation produces a write key k_t and value v_t. Each slot s has a learned slot key k_s. Write scores are computed as

$$\ell_{s,t} = \langle k_s, k_t \rangle + b_{s,t},$$

where $b_{s,t}$ optionally includes positional bias (e.g., ALiBi).

Rather than normalizing over tokens or slots globally, ASA maintains for each slot a running log-sum-exp normalizer and a weighted value sum. The slot state at time t is therefore

$$\text{slot}_s(t) = \frac{\sum_{\tau \le t} \exp(\ell_{s,\tau})\, v_\tau}{\sum_{\tau \le t} \exp(\ell_{s,\tau})}.$$

This update is implemented incrementally using a numerically stable scan and supports arbitrarily long sequences. Importantly, slot states evolve monotonically: no slot is ever reset, overwritten, or gated against its previous value.

Interpretation. The write operation performs a causal soft clustering of past tokens into a fixed number of slots. Each slot represents a smooth, convex summary of the subset of tokens that routed to it. This construction is conservative by design: information is accumulated but never selectively removed at write time.

2.2 Read and Refinement

At each timestep, a read query attends over slot states to produce a first-order readout. Optionally, a content-based term allows the query to attend directly to slot contents, weighted by a learned scalar. This first-order read remains a convex combination of slot states and therefore inherits their conservativeness.

ASA augments this with a second-order *slotspace refinement* mechanism. Refinement operates over the read weights themselves, producing a corrective signal that is added to the first-order output through a learned gate. As we show later, this refinement does not primarily add new information; instead, it selectively suppresses or reshapes components of the first-order read.

3 Training Setup and Performance

We evaluate ASM on standard language modeling benchmarks to establish empirical credibility prior to mechanistic analysis. All models are trained using teacher-forced next-token prediction with identical optimization settings across comparisons.

3.1 Datasets

We report results on:

- **WikiText-103 (raw)**: a widely used benchmark for medium-scale language modeling.
- **FineWeb (sample)**: a large-scale web corpus used to assess scalability and robustness.

All experiments use a maximum sequence length of 1024 tokens. No position extrapolation beyond the training context is performed.

3.2 Model Scales

We evaluate ASM at two representative scales:

- **57M parameters**: comparable to small transformer baselines.
- **240M parameters**: a mid-scale regime where memory effects are more pronounced.

For each configuration, we match embedding dimension, depth, and parameter count as closely as possible when comparing against standard transformer baselines.

3.3 Baselines and Results

We compare ASM against autoregressive transformers trained under identical conditions. Table 1 shows validation perplexity on WikiText-103.

Model	Parameters	Perplexity
ASM-57M	56.71M	25.22
Transformer-42M	42.32M	27.38
Transformer-57M	*forthcoming*	*forthcoming*
ASM-240M	240M	*in progress*

Table 1: Validation perplexity on WikiText-103. ASM achieves competitive performance at the 57M scale. Matched-parameter transformer baseline (57M) is currently in training.

We emphasize that these gains are not the primary contribution of this work. Rather, they demonstrate that the architectural constraints introduced for mechanistic clarity do not compromise—and may even improve—model performance.

4 Mechanistic Analysis of Refinement

We analyze the geometry of the refinement signal to understand its functional role. Let o_t denote the first-order read at timestep t, and let Δ_t denote the refinement correction added by the slotspace mechanism. We decompose Δ_t into components parallel and orthogonal to o_t:

$$\Delta_t = \Delta_t^{\parallel} + \Delta_t^{\perp}, \quad \Delta_t^{\parallel} = \frac{\langle \Delta_t, o_t \rangle}{\|o_t\|^2} o_t.$$

Across layers and heads, we observe the following consistent properties.

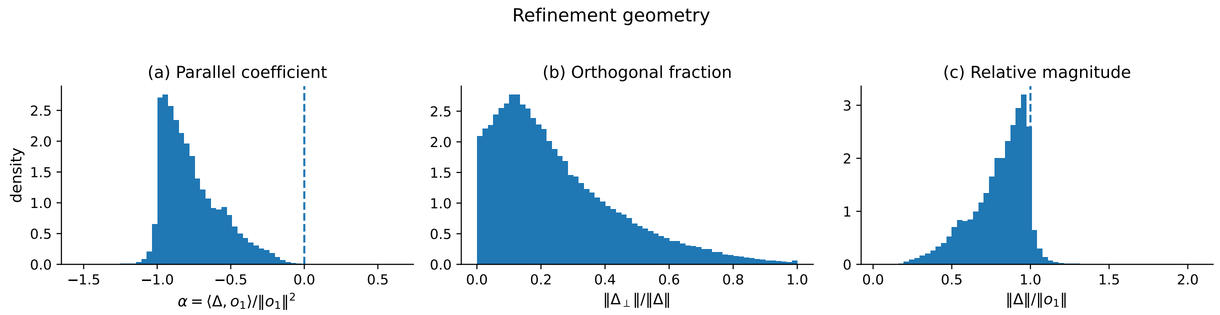

Figure 1: Geometry of the refinement signal across 200k randomly sampled positions. (a) Projection coefficient α is overwhelmingly negative (median $= -0.81$), indicating suppressive refinement. (b) Orthogonal fraction remains bounded (median $= 0.20$), with most mass below 0.3. (c) Relative magnitude stays controlled (median $= 0.84$). Vertical dashed lines indicate zero (a) and one (c) for reference.

Predominantly Anti-parallel Refinement. The projection coefficient $\alpha_t = \langle \Delta_t, o_t \rangle / \|o_t\|^2$ is almost always negative. Averaged over validation windows, layers, and heads, α_t is strongly negative, with near-zero probability mass on positive values. This indicates that refinement primarily contracts the first-order read rather than amplifying it.

Limited Orthogonal Energy. Although the orthogonal component $\Delta_t^\perp$ is non-zero, it constitutes a minority of the refinement magnitude. The fraction $\|\Delta_t^\perp\| / \|\Delta_t\|$ typically lies between 0.2 and 0.3, with a heavy tail. Thus, refinement introduces some directional adjustment, but remains dominated by suppression along the existing read trajectory.

Stable Magnitude Relative to Read. The ratio $\|\Delta_t\| / \|o_t\|$ remains bounded and typically below 1. Refinement therefore acts as a controlled correction rather than a dominant update.

Interpretation. These observations support a view of refinement as a selective suppression mechanism. The first-order read aggregates memory conservatively; refinement sharpens this aggregation by dampening most directions while selectively preserving a small subset. This explains why refinement increases effective selectivity without destabilizing memory representations.

5 Probing and Observational Analysis

Before performing causal interventions, we analyze the internal behavior of ASM to build intuition about how memory routing and refinement evolve across depth. These observations motivate the targeted interventions introduced later.

Sampling methodology. Geometric statistics are computed from 200,000 randomly sampled positions across validation batches to avoid positional bias. Random sampling reveals stronger suppression (median $\alpha = -0.81$) compared to sequential sampling (-0.72), though per-layer aggregated statistics remain identical under both methods.

5.1 Routing Composition Across Depth

We decompose slot routing signals into positional and content-based components. In early layers, routing is dominated by positional information, with slot access patterns largely determined by relative position. As depth increases, content-based routing becomes progressively more influential.

We observe a systematic shift in key–content correlation across layers: early layers exhibit weak or negative correlation, while deeper layers show increasingly positive correlation. This suggests a transition from structural organization toward semantic specialization as representations mature.

5.2 Slot Lifetimes and Memory Timescales

We estimate slot lifetimes by measuring the decay of write contributions over time. Slot write half-lives follow a heavy-tailed, approximately power-law distribution. Most slots act as short-term memory, while a small subset integrates information over hundreds of tokens.

This emergent hierarchy of timescales arises without explicit supervision or architectural differentiation, indicating that ASA naturally supports both transient and persistent memory roles.

5.3 Head and Slot Specialization

We measure effective sample size (ESS) of slot usage across layers and observe a characteristic U-shaped profile: early layers distribute attention broadly across slots, mid layers concentrate sharply, and late layers partially diffuse again.

Heads cluster into functional groups with distinct routing inertia. Importantly, we find a negative correlation between routing inertia and refinement strength: heads with more stable routing patterns tend to apply stronger refinement. This relationship suggests that refinement compensates for reduced flexibility in routing by sharpening readouts post hoc.

Together, these observations raise a central question: what computation is refinement actually performing? The following sections address this question directly.

6 Causal Interventions on Refinement Geometry

To test whether the observed refinement geometry is functionally necessary, we perform targeted interventions that selectively preserve or remove components of the refinement signal at inference time. These interventions leave all learned parameters unchanged.

6.1 Parallel vs. Orthogonal Refinement

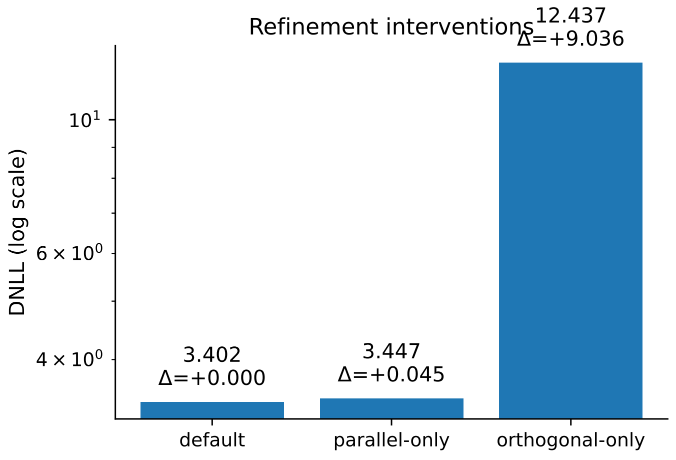

Figure 2: Causal intervention on refinement geometry. Removing the parallel component causes catastrophic degradation (DNLL increases from 3.40 to 12.44, $\Delta = +9.04$), while removing only the orthogonal component incurs minimal penalty ($\Delta = +0.045$). This demonstrates the causal necessity of suppressive refinement along the read direction.

We evaluate three conditions:

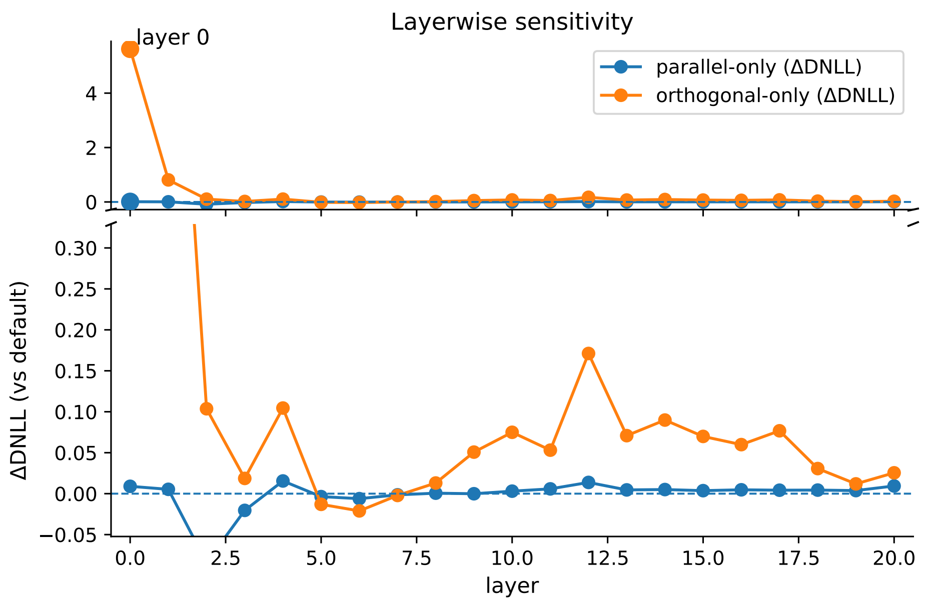

Figure 3: Layerwise sensitivity to refinement interventions. Top: full view showing catastrophic failure when removing parallel refinement in layer 0. Bottom: zoomed view reveals that parallel-only refinement incurs minimal degradation across all layers, while orthogonal-only refinement shows decreasing but persistent degradation in deeper layers. Layer 0 exhibits extreme sensitivity, consistent with early-layer conservative memory aggregation.

- **Default:** full refinement Δ_t applied.
- **Parallel-only:** $\Delta_t^{\parallel}$ applied, discarding $\Delta_t^{\perp}$.
- **Orthogonal-only:** $\Delta_t^{\perp}$ applied, discarding $\Delta_t^{\parallel}$.

Removing the parallel component significantly degrades performance. On WikiText-103 validation data, orthogonal-only refinement increases negative log-likelihood catastrophically (from ≈ 3.4 to > 12). In contrast, parallel-only refinement incurs only a modest degradation and in some layers slightly improves performance.

6.2 Layerwise Interventions

Applying orthogonal-only refinement at individual layers reveals that early layers are particularly sensitive: disabling the parallel component in the first layer produces the largest increase in loss. This aligns with earlier findings that early layers rely more heavily on positional routing and conservative memory aggregation.

6.3 Orthogonal Scaling Sweep

We further scale the orthogonal component by a factor β. Performance improves slightly for small $\beta > 0$, indicating that a limited orthogonal adjustment is beneficial. However, increasing β beyond this range leads to rapid degradation, confirming that excessive orthogonal correction destabilizes the readout.

Conclusion. These interventions establish that refinement is not merely an auxiliary attention mechanism. Its dominant parallel, suppressive component is causally necessary for good performance, while the orthogonal component plays a secondary, tightly constrained role. Together with the mechanistic analysis, this supports the interpretation of ASA's refinement as a controlled sharpening operation acting on conservative memory reads.

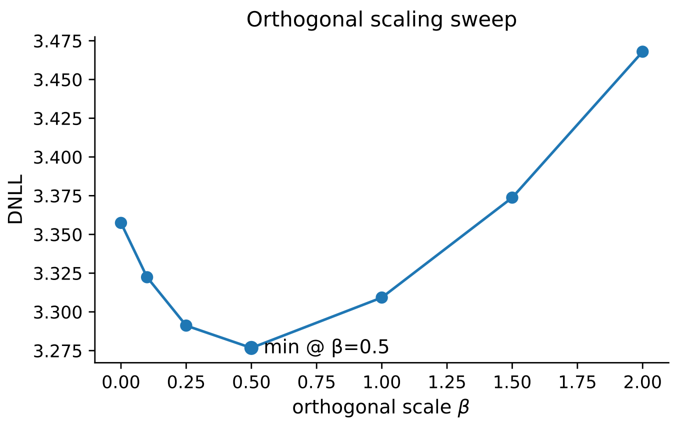

Figure 4: Effect of scaling the orthogonal refinement component by factor β. Performance is optimal at $\beta = 0.5$ (DNLL = 3.28), indicating that a moderate amount of orthogonal adjustment is beneficial. However, both removing orthogonal refinement entirely ($\beta = 0$, DNLL = 3.36) and excessive scaling ($\beta = 2$, DNLL = 3.47) degrade performance, confirming the tightly constrained role of orthogonal correction.

7 Limitations

Our work has several limitations.

First, ASM does not extrapolate reliably beyond its training context length. Although slot memory persists indefinitely in principle, positional biases constrain effective generalization beyond the trained sequence length.

Second, we focus exclusively on language modeling and do not evaluate instruction-following or downstream tasks. It remains an open question how refinement behaves under supervised fine-tuning.

Third, while we analyze models up to 240M parameters, we do not explore very large-scale regimes. Scaling behavior of refinement at billions of parameters remains to be studied.

Finally, our mechanistic analysis is limited to the refinement operation. While ASA enables clearer intervention than standard attention mechanisms, fully characterizing the interaction between routing, writing, and refinement remains an open challenge.

8 Related Work

ASM draws inspiration from several lines of prior work.

Memory-Augmented Models. Architectures such as Transformer-XL, Compressive Transformers, and external memory networks extend attention with persistent state. Unlike these approaches, ASM constructs memory through monotonic aggregation and introduces an explicit refinement stage, enabling causal intervention on memory usage.

Recurrent and State-Space Models. Recurrent neural networks and modern state-space models maintain persistent hidden state but typically lack explicit addressing and interpretable access mechanisms. ASM combines persistent state with addressable, slot-based structure through its ASA layers.

Attention Mechanisms. Standard self-attention entangles retrieval and selection in a single softmax. ASA separates these roles, allowing retrieval to remain conservative while refinement performs selective suppression.

Mechanistic Interpretability. Recent work on attention analysis and causal tracing has highlighted the difficulty of intervening cleanly in transformer internals. ASA is designed to be intervenable by construction, enabling direct causal tests of hypothesized mechanisms.

9 Conclusion

We introduced Addressed State Memory, an architecture that separates conservative memory construction from selective memory utilization through an explicit refinement operation. While ASM achieves competitive language modeling performance, its primary contribution lies in mechanistic clarity.

Through geometric analysis and causal intervention, we show that ASA's refinement functions as directional suppression rather than additive feature synthesis. Its parallel, contractive component is causally necessary for stable computation, while its orthogonal component plays a constrained, depth-dependent role.

More broadly, this work suggests that designing architectures with explicit, intervenable structure can both improve performance and enable deeper understanding of learned computation. We view ASM as a step toward models whose internal mechanisms are not only powerful, but also scientifically legible.

References

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *NeurIPS*.

[2] Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. V., and Salakhutdinov, R. (2019). Transformer-XL: Attentive language models beyond a fixed-length context. In *ACL*.

[3] Press, O., Smith, N. A., and Lewis, M. (2022). Train short, test long: Attention with linear biases enables input length extrapolation. In *ICLR*.

[4] Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Johnston, S., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. (2022). In-context learning and induction heads. *Transformer Circuits Thread*.

[5] Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., DasSarma, N., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. (2021). A mathematical framework for transformer circuits. *Transformer Circuits Thread*.