EMMA: Equilibrium Memory for Mamba/Liquid Agents

Equilibrium Memory Research Lab (EMRL), Phoenix, AZ
equilibriummemory@gmail.com

September 15, 2025

Abstract

We introduce EMMA, a streaming sequence architecture that fuses a compact state-space backbone with a vector-symbolic associative memory inside a deep-equilibrium (DEQ) fixed point. Each step performs a single equilibrium solve that reconciles current evidence with episodic recall; training uses implicit differentiation, keeping activation memory flat in sequence length. The EAMES variant augments memory with lightweight error-correcting codes, locality buckets, equivariant keys, and Sinkhorn addressing. On long-range synthetic probes (e.g., Needle) EMMA converges in ˜5-8 solver iterations per step and exhibits causal memory use (ablations with no-write and shuffled reads reduce both validation accuracy and top-k hit rate). Headline**.** On NIH, a pinned n=2 model reaches seed-averaged validation accuracy **0.788** by epoch 1 with $\bar{K} \approx 8$ fixed-point iterations. A 1-epoch CPU probe shows +**1.8**% tokens/sec with **12.6**% lower peak RAM (Table **??**). A larger n=4, L=512 scaling probe attains 0.083 ± 0.043 across two seeds, highlighting variance to address at higher capacity.

1 Introduction

The cost of quadratic attention scales poorly with long contexts, while fixed windows discard information. Linear-time sequence models such as structured state-space models (SSM) and Mamba reduce complexity, but a finite hidden state constrains long-term recall [11, 10]. External memory approaches demonstrate content-addressable retrieval [8, 9, 15], yet often introduce controller complexity and multi-step interactions at inference. We propose EMMA, which *fuses* a streaming backbone with a vector-symbolic memory (VSA) inside a DEQ fixed point. The equilibrium solve plays the role of depth: it iteratively reconciles evidence from the current token with recall from memory, yielding a single latent $\mathbf{z}^*$ that is consistent with both. By training through equilibria [1, 2], EMMA keeps activation memory essentially constant in sequence length, while providing explicit, interpretable memory diagnostics.

Contributions.

1. **Fixed-point fusion.** A single DEQ solve per step integrates SSM/Liquid features with VSA recall, enabling streaming inference with constant training memory.
2. **EAMES** memory upgrades**.** We incorporate ECC coding, locality buckets, equivariant keys, and Sinkhorn addressing to improve robustness under superposition [16, 12, 14, 4, 7].
3. **Practical recipe and diagnostics**. A stabilized write curriculum (oracle → mixed → learned) and solver caps yield reliable training; we log read/write cosine, top-k hit-rate, and fixed-point iterations for transparency.

4. **Empirics.** On long-range probes, EMMA converges in ˜5-8 iterations and shows causal memory use via ablations (§**??**). On the pinned n=2 setting we observe seed-averaged validation accuracy of 0.788 by epoch 1 with $\bar{K} \approx 8$ fixed-point iterations; as a scaling probe, n=4, L=512 attains 0.083 ± 0.043 across two seeds. A 1-epoch CPU probe shows +1.8% tokens/sec with 12.6% lower peak RAM (Table **??**).

2 Related Work

Equilibrium and implicit layers. DEQ models treat depth as a fixed point, enabling constant-memory training and flexible solver budgets [1, 2]. EMMA uses a single equilibrium solve to fuse evidence with memory recall. Linear-time sequence models**.** SSM models and MAMBA deliver strong long-range modeling with linear complexity [11, 10, 6], but capacity remains bounded by hidden state. Efficient attention**.** IO-aware attention [5], linear/performer variants [13, 3, 19], and convolutional hybrids [17] trade constants and inductive biases for range. Associative memory and VSA**.** HRR and hyperdimensional computing provide algebra for binding/bundling with simple cleanup [16, 12, 14]. External memory controllers. NTM/DNC and modern Hopfield networks demonstrate content addressing and high capacity [8, 9, 15, 18]. EMMA differs by solving a single equilibrium that *jointly* reconciles backbone dynamics with memory reads.

3 The EMMA/EAMES Architecture

We summarize the main elements; a formal system description with notation appears in the internal technical note (in preparation).

3.1 Streaming backbone and keying

A small SSM/LIQUID backbone encodes $(x_t, \mathbf{h}_{t-1})$ to a normalized key $\mathbf{k}_t \in \mathbb{R}^D$ and a proposal latent. Keys are ℓ_2-normalized; group-equivariant parameterizations are optional in EAMES.

3.2 Vector-symbolic memory

We use HRR-style binding $\otimes$ and unbinding $\oslash$, with superposition $\oplus$ and a lightweight learned cleanup module; the memory state $\mathbf{M}$ can be sharded into locality buckets. Reads compute $\mathbf{r}_t = \mathrm{unbind}(\mathbf{M}, \mathbf{k}_t)$ (or a Sinkhorn-weighted mixture across buckets [4]); writes add bound traces with optional ECC coding for robustness.

3.3 Equilibrium fusion and training

Define a contractive residual map $F(\mathbf{z}; x_t, \mathbf{r}_t) = f_\theta(\mathbf{z}; x_t, \mathbf{r}_t) - \mathbf{z}$. We solve $\mathbf{z}^* = \arg\min_{\mathbf{z}} \|F(\mathbf{z}; x_t, \mathbf{r}_t)\|$ (Anderson/Broyden) with an iteration cap during training. Gradients use implicit differentiation through $\mathbf{z}^*$ [1].

Algorithm 1 EMMA step (read → fixed point → optional write)

Require: input x_t, previous state $\mathbf{h}_{t-1}$, memory $\mathbf{M}$

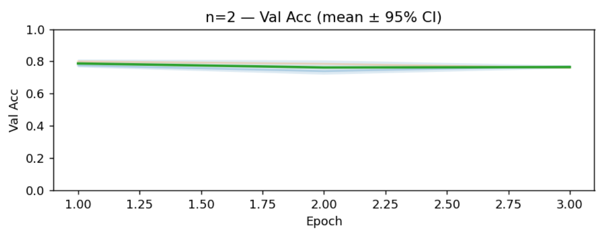
$\mathbf{k}_t \leftarrow \text{normalize}(g_\theta(x_t, \mathbf{h}_{t-1}))$ ▷ keyer (L2 norm)
if locality buckets **then**
  $\mathcal{C} \leftarrow \text{bucketize}(\mathbf{k}_t)$; $\alpha \leftarrow \text{Sinkhorn}(\text{score}(\mathbf{k}_t, \{M_b\}))$
  $\mathbf{r}_t \leftarrow \sum_{b \in \mathcal{C}} \alpha_b \cdot \text{unbind}(M_b, \mathbf{k}_t)$
else
  $\mathbf{r}_t \leftarrow \text{unbind}(\mathbf{M}, \mathbf{k}_t)$ ▷ e.g., HRR correlation + cleanup
end if
$\mathbf{z}^* \leftarrow \text{solve}\{\mathbf{z} = f_\theta(\mathbf{z}; x_t, \mathbf{r}_t)\}$ ▷ Anderson/Broyden; capped iters
$\mathbf{h}_t \leftarrow u_\theta(\mathbf{h}_{t-1}, \mathbf{z}^*)$; $y_t \leftarrow o_\theta(\mathbf{z}^*)$
if gate$(x_t, \mathbf{z}^*)$ **then**
  $v \leftarrow \text{encode}(\mathbf{z}^*)$; $v \leftarrow \text{ECC}(v)$ ▷ EAMES: optional
  $\mathbf{M} \leftarrow \alpha\mathbf{M} \oplus \beta\,\text{bind}(\mathbf{k}_t, v)$ ▷ decay α, write scale β
end if
return $y_t, \mathbf{h}_t, \mathbf{M}$

4 Training & Implementation

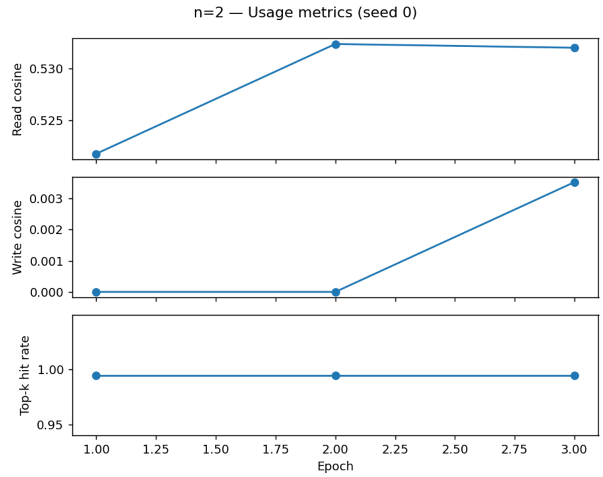
Fixed-point and schedule. We cap DEQ iterations at 8 during training and monitor both residual norms and mean fixed-point iterations. A three-stage write curriculum stabilizes learning: warm-start for two epochs, then a ramp (four to six epochs) to a mixing floor of 0.3. When `mem_into_deq` is enabled, we schedule a small memory scale (0.5) through the ramp.

Optimization and logging. We use AdamW (lr 3×10^{-3}, wd 10^{-2}, batch 32) with a post-warm learning-rate factor of 0.5. Each epoch logs validation accuracy, `avg_fp_iters`, and memory diagnostics: last-step read/write cosine and top-k hit rate. All key hyperparameters appear in **??**; configs are in the appendix bundle. We examine n=4 at L=512 as a capacity/length probe. Across two seeds we observe final val acc 0.125 and 0.040 (mean 0.083 ± 0.043); ablations suggest memory reads are causal (see causality bars). With a short write curriculum (warm=2, ramp=4-6, floor=0.3) the pinned n=2 model settles in ∼5-8 iterations. we observe best seed-averaged validation accuracy **0.788** by epoch 1 with $\bar{K} \approx 8$ fixed-point iterations.

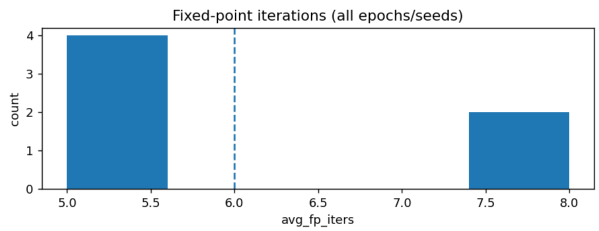
Condition	Val Acc ↑	Read Cos ↑	Top-k Hit ↑
Normal	0.793 ± 0.016	0.525 ± 0.008	0.997 ± 0.003
Eval–NoWrite	0.020 ± 0.005	0.000 ± 0.000	0.330 ± 0.000
Eval–ShuffleRead	0.637 ± 0.054	0.475 ± 0.024	0.870 ± 0.028



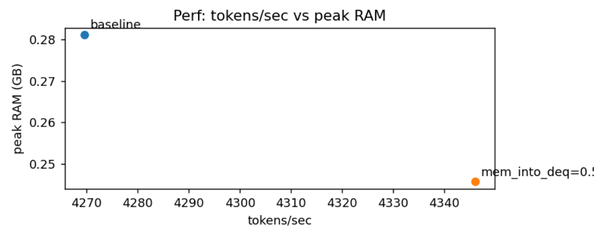
(a) n=2: val acc, mean±95% CI (per-seed faint).



(b) n=2: usage metrics by epoch (stacked panels).

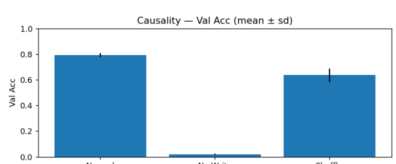


(c) Fixed-point iterations distribution.

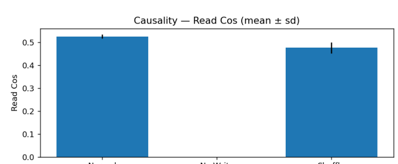


(d) Perf: tokens/sec vs peak RAM (GB).

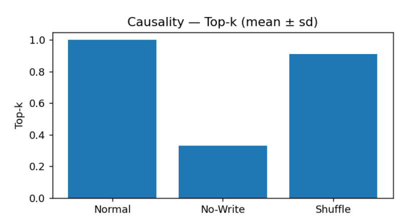
Figure 1: Richer summary of learning dynamics, solver behavior, and efficiency.



(a) Val Acc



(b) Read Cos



(c) Top-k

Figure 2: **Causality ablations (n=2).** Disabling writes (*Eval-NoWrite*) collapses performance; shuffling read keys (*Eval-ShuffleRead*) degrades both accuracy and usage metrics. Error bars are mean±sd over three seeds. See Table 1 for exact values.

5 Discussion

When the fixed point helps. The DEQ solve integrates noisy reads with current evidence, reducing the need for deep stacks while keeping training memory flat [1]. Gains are most pronounced when reads are ambiguous or aliased. Capacity and interference. VSA superposition admits algebraic capacity analysis; EAMES mitigates interference via ECC and locality. Latency and stability. Solver latency spikes can occur for difficult tokens; iteration caps and damping stabilize training at small cost to accuracy.

6 Limitations

EMMA relies on a contractive residual map and careful scheduling; pathological regimes can stall convergence. Memory interference may accumulate under heavy load. Our evaluation is focused on synthetic probes; broader downstream tasks and larger-scale pretraining remain future work. Our n=4 runs exhibit high variance across seeds (0.083 ± 0.043, two seeds), suggesting training stability at larger capacity/length is an important target for future work.

7 Conclusion

EMMA unifies streaming dynamics, algebraic memory, and equilibrium inference. Early results indicate promising long-context behavior with interpretable recall. Future directions include multimodal adapters, KV-cache augmentation, and larger-scale pretraining.

Reproducibility. Configs for all experiments appear in the appendix bundle; we log seeds and provide CSVs of accuracy, fixed-point iterations, and usage metrics. Environment and commit: Python 3.11, git commit N/A (local workspace). Artifact URL: `https://github.com/equilibriummemory-cmyk/EMMA`. Use of AI tools. The writing and tooling for this project made use of an AI assistant as a drafting and engineering aid (figure scripting, LaTeX packaging, and run orchestration). The assistant is not listed as an author and did not make independent scientific claims; all experiments and final edits were conducted and approved by the human author.

References

[1] Shaojie Bai, Vladlen Koltun, and J Zico Kolter. Deep equilibrium models. In *NeurIPS*, 2019.

[2] Shaojie Bai, Vladlen Koltun, and J Zico Kolter. Multiscale deep equilibrium models. *NeurIPS*, 2021.

[3] Krzysztof Choromanski et al. Rethinking attention with performers. In *ICLR*, 2021.

[4] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *NeurIPS*, 2013.

[5] Tri Dao et al. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *NeurIPS*, 2022.

[6] Tri Dao et al. Hungry hungry hippos: Towards language modeling with state space models. In *ICLR*, 2024.

[7] Robert G Gallager. *Low-Density Parity-Check Codes*. PhD thesis, MIT, 1963.

[8] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. In *arXiv:1410.5401*, 2014.

[9] Alex Graves et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 2016.

[10] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv:2312.00752*, 2023.

[11] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *ICLR*, 2022.

[12] Pentti Kanerva. Hyperdimensional computing: An introduction to computing in distributed representation. *Cognitive Computation*, 2009.

[13] Angelos Katharopoulos et al. Transformers are rnns: Fast autoregressive transformers with linear attention. In *ICML*, 2020.

[14] Denis Kleyko et al. Vector symbolic architectures and hyperdimensional computing: A review. *Journal of Artificial Intelligence Research*, 2022.

[15] Dmitry Krotov and John J Hopfield. Large associative memory problem in neural networks. *arXiv:2008.06996*, 2020.

[16] Tony A Plate. Holographic reduced representations. In *ICANN*, 1995.

[17] Stanislas Polu et al. Hyena hierarchy: Towards larger context with efficient convolutions. In *ICML Workshop*, 2023.

[18] Johannes Ramsauer et al. Hopfield networks is all you need. In *ICLR*, 2021.

[19] Imanol Schlag et al. Linear transformers are secretly fast weight programmers. In *ICLR*, 2021.

A Additional Details and Configs

Extended derivations, full algorithms, and YAMLs. Additional plots include mem_scale sweeps for `mem_into_deq` and an A/B seed study; see the artifact bundle for CSVs and scripts.