Neuromodulatory-Dynamic State Machine (N-DSM): A Unified Architecture Integrating Stateful Recurrence, Dynamic Gating, and Meta-Learned Control (Research Proposal)

Author: Michael Morgan
Date: October 15, 2025

Abstract

The dominant Transformer architecture for large language models (LLMs), while powerful, is fundamentally a static, feed-forward function that exhibits significant limitations in tasks requiring stateful reasoning, long-range causal consistency, and computational efficiency. Empirical evidence from cognitive benchmarks and long-form dialogue analysis reveals a lack of robust working memory and a failure to maintain a persistent, evolving internal state. To address these deficiencies, we propose the **Neuromodulatory-Dynamic State Machine (N-DSM)**, a novel, brain-inspired architectural paradigm that replaces static computation with a dynamic, multi-scale learning process. The N-DSM is built on a synthesis of principles from neuroscience, meta-learning, and modern efficient sequence models.

Its core component is the **Dynamic State Propagation (DSP)** block, a stateful, recurrent module designed to replace the standard Transformer block. The DSP block leverages a State Space Model (SSM) backbone, inspired by architectures like Mamba and RetNet, to achieve linear-time complexity and a continuously evolving hidden state that models the "lingering" of information. This temporal processing is augmented with dynamic, cross-layer connections inspired by MUDDFormer, allowing for adaptive hierarchical information flow. Crucially, the state update mechanism is not a simple transformation but an explicit, meta-learned optimization step, framed as a confidence-guided delta rule inspired by the theories of Fast Weight Programmers and mesa-optimization.

Overseeing this system is a parallel **Neuromodulatory Controller (NMC)**, a small meta-network that models the function of the prefrontal cortex and global neuromodulation. The NMC observes the aggregate state of all DSP blocks and broadcasts a low-dimensional control vector that dynamically alters the core operational parameters of the primary network—such as its learning rates, decay constants, and gating mechanisms—in a context-sensitive manner. This allows the N-DSM to learn to regulate its own plasticity and computational state in response to task demands and internal uncertainty. By unifying recurrent dynamics, fast weight plasticity, and global neuromodulatory control, the N-DSM offers a principled path toward a new class of LLMs that are more efficient, stateful, and fundamentally aligned with the computational strategies of biological intelligence.

Part 1: Conceptual Framework and Core Principles

1.1 The Fundamental Problem: The Static, Stateless Nature of Transformers

The current dominant paradigm in large language models, the Transformer architecture (as formalized in Phuong & Hutter, "Formal Algorithms for Transformers"), suffers from a core conceptual limitation: it is a fundamentally static, feed-forward function. At each forward pass, it re-computes its representation of the world from scratch based on its input context window. This leads to several well-documented failures:

- **Lack of a "Mental Scratchpad":** As empirically demonstrated in "Working Memory Identifies Reasoning Limits in Language Models," Transformers struggle with tasks like the n-back benchmark that require the active maintenance and manipulation of information not explicitly present in the immediate context. They lack a robust, dynamic working memory.
- **Failure in Long-Range Causal Reasoning:** The paper "Evaluating Very Long-Term Conversational Memory of LLM Agents" shows that even with massive context windows or Retrieval-Augmented Generation (RAG), models fail to maintain causal and temporal consistency over long dialogues. Their "memory" is a form of stateless search, not a stateful understanding of an evolving narrative.
- **Computational Inefficiency:** The quadratic complexity of the attention mechanism, while powerful, makes processing truly long sequences computationally infeasible, a problem that has spawned an entire field of "x-former" variants trying to achieve linear-time complexity.

1.2 A Brain-Inspired Solution: Three Core Principles of Dynamic Computation

Our proposed N-DSM architecture is a direct response to these limitations, drawing its core principles from the biological and theoretical papers in our context window. It is built on the understanding that intelligence in dynamic environments (like conversation or reasoning) is not a static computation but an active, stateful process. The architecture is founded on three pillars:

- **Principle 1: Continuous Statefulness and Reverberating Activity.** Your initial intuition was key: thoughts "linger." Biological neurons don't reset after each firing; their chemical state persists and influences subsequent computation. The **Maelstrom Networks** paper provides a computational analog with its recurrent, reverberating core. The N-DSM moves away from the "memoryless" paradigm and embraces the necessity of a persistent, continuously evolving internal state that represents the model's "current understanding" or "train of thought."

- **Principle 2: Dynamic, Context-Aware Computation via Fast Weights.** The brain is not a fixed circuit; it constantly re-wires itself on short timescales. This is the "Trifecta of Fast Weights" we identified:
    1. The *concept* of a "slow" network learning to program the "fast" weights of another network (**Schmidhuber, 1991**).
    2. The *biological implementation* via short-term synaptic plasticity, where a neuron's synapses are transiently potentiated based on signal coincidence (**Ellwood, "Short-term Hebbian learning..."**).
    3. The *theoretical justification* that this synaptic plasticity is the mathematically optimal solution for inference in continuously changing environments (**"Optimality of short-term synaptic plasticity..."**).
       The N-DSM will replace the static attention mechanism with a dynamic one based on this principle, where the effective weights of the network are modified "on the fly" as part of the forward pass itself.
- **Principle 3: Hierarchical and Neuromodulated Control.** In the brain, high-level cognitive functions, like attention and goal-setting, are not managed by the same circuits that process raw sensory data. The Prefrontal Cortex (PFC) sends global, modulatory signals to other brain regions. The paper **"Prefrontal Cortex Encodes Value Pop-out..."** shows this is a rapid process (~150ms) tied to the *value* or relevance of a target. Furthermore, the **"Neuromodulation enhances dynamic sensory processing..."** paper shows how a secondary network can learn to dynamically alter the core computational parameters (e.g., neuronal gain, time constants) of a primary network. The N-DSM will incorporate this by having a separate, smaller meta-controller that oversees the state of the primary network and adjusts its behavior based on the high-level context of the task.

1.3 The N-DSM Architecture at a Glance

To realize these principles, the N-DSM is composed of two main components, which will be detailed in the following sections:

1. **The Dynamic State Propagation (DSP) Block:** This is the workhorse of the model, replacing the standard Transformer block. It combines the linear-time efficiency of modern State Space Models (SSMs) like Mamba with dynamic, cross-layer connections inspired by MUDDFormer, allowing information to be processed both sequentially (in time) and hierarchically (through depth) in a data-dependent way.
2. **The Neuromodulatory Controller (NMC):** This is a small, parallel meta-controller that observes the global state of all DSP blocks and learns to broadcast a low-dimensional control vector back to them. This vector dynamically adjusts the core operational parameters

of the DSP blocks, implementing a form of system-wide, learned attention and adaptive control.

In essence, the DSP blocks are the "cortex," processing information with a lingering, dynamic state. The NMC is the "prefrontal cortex/neuromodulatory system," guiding what the cortex should pay attention to and how it should process it. This composite system is designed to be a stateful, efficient, and biologically-plausible alternative to the Transformer.

Part 2: The Dynamic State Propagation (DSP) Block

2.1 Overview: A Hybrid, Stateful, and Hierarchical Processor

The Dynamic State Propagation (DSP) Block is the fundamental building block of the N-DSM, designed as a drop-in replacement for the standard Transformer block. Its primary function is to address the dual limitations of the Transformer: its quadratic complexity in sequence length and its stateless, feed-forward nature. The DSP Block achieves this by hybridizing three key concepts from the provided research:

1. A **State Space Model (SSM) Backbone** for efficient, linear-time recurrent processing of sequences.
2. **Dynamic Hierarchical Gating** for adaptive, cross-layer information flow, breaking the bottleneck of simple residual connections.
3. An **Adaptive Delta Rule Update** at its core, reframing the state transition as an explicit, learned optimization process.

2.2 Component 1: The State Space Model (SSM) Backbone for Temporal Dynamics

The foundation of the DSP Block is a modern State Space Model. This directly implements **Principle 1 (Continuous Statefulness)**.

- **Architectural Choice:** We select the **Mamba** architecture (Gu & Dao, "Mamba: Linear-Time Sequence Modeling with Selective State Spaces") as our primary SSM backbone. Unlike traditional RNNs which suffer from vanishing gradients, and unlike Transformers which are stateless, Mamba is based on the **Structured State Space (S4)** framework (Gu, Goel, & Ré, "Efficiently Modeling Long Sequences..."). An SSM is a continuous-time system discretized for computation, making it the ideal mathematical formalization for a "lingering thought" with a decaying state.
- **Mechanism:** Mamba models a sequence by updating a hidden state `h` via the recurrence `h_t = f(h_{t-1}, x_t)`. Its crucial innovation is the "selective" mechanism, where the state transition matrices (`A`, `B`) become dynamic functions of the input token `x_t`. This allows the model to learn, on-the-fly, whether to remember the current input (by strengthening its influence on the state) or to ignore it and persist the previous state. This is a powerful, learned form of input-dependent gating.
- **Addressing Limitations:** We explicitly acknowledge the critique from **"The Illusion of State in State-Space Models"** (Merrill et al.). This paper proves that current SSMs like Mamba, while efficient, are computationally limited (within the complexity class TC^0) and cannot solve

certain complex, state-tracking problems. This suggests that the fixed-size state of a pure SSM, while powerful, is not a complete solution. Therefore, in the N-DSM, the SSM backbone is not used in isolation; it serves as an efficient and powerful foundation upon which more complex dynamics are built. The recent **MemMamba** paper (Wang et al.), which adds an explicit "note-taking" memory to the standard Mamba state, provides a compelling precedent for this hybrid approach, creating a dual-memory system.

2.3 Component 2: Dynamic Hierarchical Gating for Cross-Layer Communication

To overcome the limitations of a simple stack of recurrent blocks, the DSP integrates a dynamic routing mechanism for information flow *through the depth of the network*.

- **Architectural Choice:** We replace the standard residual connection (`output = input + Block(input)`) with the **MUltiway Dynamic Dense (MUDD)** connection mechanism from **"MUDDFormer"** (Xiao et al.).
- **Mechanism:** In a standard deep model, the input to layer `L` is simply the output of layer `L-1`. With MUDD, the input to layer `L` is a *dynamically weighted sum* of the outputs from *all preceding layers* (`0` to `L-1`). The weights for this summation are not fixed; they are generated by a small MLP at layer `L` that takes the current hidden state `x_t` as input. This allows the network, for each token, to decide which previous layers are most relevant and create a direct "shortcut" to pull information from them.
- **Synergy with SSM:** By combining Mamba with MUDD, the N-DSM can process information dynamically in two dimensions. Mamba manages the state evolution *temporally* along the sequence, while MUDD manages information flow *hierarchically* across layers. For example, when processing a complex sentence, a Mamba block at layer 15 might learn that for the current word, the most useful context is not the output of layer 14, but a combination of the raw embedding (layer 0) and a mid-level feature from layer 7. This is a direct parallel to the **"Dendritic Cortical Circuits"** paper (Wybo et al.), where top-down contextual signals (analogous to MUDD connections from higher layers) modulate the processing of bottom-up signals.

2.4 Component 3: The Adaptive Delta Rule for State Updates

The heart of the DSP Block is how its state evolves. We reframe the recurrent update not as a simple transformation, but as a meta-learned, single-step optimization, directly realizing **Principle 2 (Fast Weights)**.

- **Theoretical Foundation:** We build on the thesis from **"Transformers learn in-context by gradient descent"** (von Oswald et al.), which posits that the forward pass is an implicit optimization. We make this explicit.
- **Mechanism:** The state update of our Mamba-like backbone will be formulated as a delta rule, inspired by the **"Self-Referential Weight Matrix"** (Irie, Schmidhuber et al.). The update to the state `h` will take the form:
  `h_t = h_{t-1} + β_t * (V_t - h_{t-1} * K_t)`
  Here, `K_t` (key) and `V_t` (value) are derived from the current input `x_t`. The crucial component is the learning rate `β_t`.
- **Confidence-Guided Learning Rate:** `β_t` is not a fixed or simply projected value. Inspired by the **TTT3R** paper (Chen et al.), `β_t` will be a *confidence-guided* learning rate. It will be computed based on the "match" or alignment between the current input `x_t` (which generates the query) and the current memory state `h_{t-1}` (which is probed). A high-confidence match leads to a larger `β_t`, allowing the new information to strongly update the state. A low-confidence match (e.g., encountering an unexpected token) results in a smaller `β_t`, preserving the old state and preventing catastrophic forgetting. This mechanism ensures that the model learns to trust its own memory, updating it deliberately rather than being overwritten at every step.

In summary, the DSP Block is a sophisticated recurrent module that processes sequences efficiently using an SSM backbone, routes information intelligently across layers using dynamic gating, and updates its internal state via a meta-learned, confidence-guided delta rule. It is designed to be a stateful, adaptive, and scalable replacement for the Transformer block.

Part 3: The Neuromodulatory Controller (NMC)

3.1 Overview: A System for Global, Adaptive Control

While the Dynamic State Propagation (DSP) Blocks provide powerful, stateful processing at the local and hierarchical levels, the Neuromodulatory Controller (NMC) introduces a global, top-down control mechanism. This directly implements **Principle 3 (Hierarchical and Neuromodulated Control)**.
The NMC is a separate, smaller network that runs in parallel to the main DSP stack. Its function is not to process the *content* of the sequence, but to monitor the *state of the computation* and broadcast low-dimensional control signals that dynamically alter the operational parameters of all DSP blocks.

This design is heavily inspired by two biological concepts:

1. **Top-Down Attentional Control:** The brain's prefrontal cortex doesn't process every detail; it directs other sensory and cognitive areas to focus on what is currently relevant or valuable. The paper **"Prefrontal Cortex Encodes Value Pop-out in Visual Search"** (Abbaszadeh et al.) provides a powerful example, showing that the PFC rapidly (~150ms) signals the presence of a high-value target, allowing for an efficient "pop-out" search that bypasses slow, serial processing. The NMC is designed to be the architectural analog of this value-driven executive function.
2. **Neuromodulation:** The brain's chemical environment, controlled by neuromodulators like acetylcholine and dopamine, can globally change the computational properties of entire neural populations. The paper **"Neuromodulation enhances dynamic sensory processing in spiking neural network models"** (AlKilany & Goodman) demonstrates this computationally, showing that a secondary network can learn to dynamically adjust parameters like neuronal gain and time constants in a primary network to improve performance on challenging tasks. The NMC learns to generate these neuromodulatory signals.

3.2 Architecture of the Neuromodulatory Controller (NMC)

The NMC is designed for efficiency and global context awareness.

- **Input to the NMC:** The NMC does not see the raw input tokens `x_t`. Instead, its input at each time step `t` is a compressed summary of the entire network's state. This is constructed by gathering the hidden states `h_t` from *every* DSP Block at the current step and passing them through a pooling or attention mechanism. This gives the NMC a holistic, real-time view of the main network's activity.

- **NMC Model:** The NMC itself is a small, fast recurrent model. A gated recurrent unit (GRU) or a small Mamba-style SSM is a suitable choice. Its task is not to model the full complexity of language but to learn the dynamics of the computational state itself. Its hidden state represents a summary of the network's processing trajectory.
- **Output of the NMC:** The NMC outputs a low-dimensional "neuromodulatory vector" `n_t` at each time step. This vector is intentionally kept small (e.g., 16 to 64 dimensions) to represent a global, broadcasted signal, analogous to the diffuse action of chemicals in the brain.

3.3 The Mechanism of Neuromodulation: Dynamic Parameter Control

The key innovation is how the neuromodulatory vector `n_t` is used. It is **not** added to the residual stream. Instead, it is used to dynamically alter the core operational parameters within each DSP Block across the entire network.

This is achieved by making key parameters of the DSP Blocks functions of `n_t`. For each DSP block at layer `L`, its parameters for time step `t` are computed as: `Param_L(t) = f_L(BaseParam_L, n_t)`, where `f_L` is a small, layer-specific MLP.

The NMC learns to control several critical parameters, including:

- **The Delta Rule Learning Rate (** The NMC provides a global, task-level signal that modulates the locally computed, confidence-guided learning rate within each DSP Block's state update rule. This creates a two-level control system: the local mechanism prevents catastrophic forgetting on a step-by-step basis, while the global NMC can, for example, learn to increase the overall plasticity of the network when it detects a new topic is being introduced, or decrease it to consolidate memory during a stable context.
- **SSM Decay/Transition Parameters:** The neuromodulatory vector can directly alter the core dynamics of the SSM backbone. In a Mamba-like model, `n_t` would influence the Δ parameter, effectively controlling the model's focus on the current input versus its past state. In a RetNet-style model (**"Retentive Network: A Successor to Transformer..."**, Sun et al.), it could control the global decay rate γ, dictating the memory timescale for the entire network.
- **Dynamic Gating in Hierarchical Connections:** The `n_t` vector can act as a modulator for the MUDDFormer-style gates. This would allow the NMC to globally bias the network's information flow. For example, it could learn that for summarization tasks, it should encourage higher layers to draw information directly from lower layers (biasing the MUDD gates towards that configuration), while for creative generation, it might encourage more local, layer-to-layer processing.

3.4 Connection to Uncertainty and Control

This architecture provides a natural, emergent mechanism for uncertainty handling, connecting back to the ideas from **Maelstrom Networks** and **ChemAU**.

- When the main DSP network enters a state of high flux or "chaos" (e.g., due to contradictory or novel inputs), the sequence of states fed into the NMC will be highly variable and unpredictable.
- The NMC can learn to recognize these chaotic input patterns. In response, it can learn to output a neuromodulatory vector n_t that induces a more "cautious" computational state in the main network—for example, by globally reducing the delta rule learning rates (β_t), shortening memory timescales, or increasing the temperature of softmax functions in the MUDD gates.
- This is a far more elegant and powerful implementation of an uncertainty principle than post-hoc logit analysis (as in **ChemAU**). The system learns to regulate its own computational dynamics in response to its internal state of uncertainty, a hallmark of robust biological cognition. This also relates to the **"Whispering Experts"** approach, but instead of dampening specific "toxicity neurons," the NMC can learn to shift the entire network into a state that makes toxic generation less probable.

In summary, the NMC acts as the "brain" of the N-DSM. It abstracts away from the content of the sequence to reason about the process of computation itself. By learning to generate global control signals, it allows the network as a whole to adapt its behavior, manage its own plasticity, and respond intelligently to uncertainty in a unified, biologically-inspired framework.

Part 4: Learning Paradigm and Theoretical Foundations

4.1 Overview: A Multi-Scale, Meta-Learned System

The N-DSM is not merely a collection of architectural components; it is defined by its learning paradigm. Unlike a standard Transformer which is trained once and then remains static, the N-DSM is a system with three distinct timescales of learning and adaptation, each grounded in the research we have analyzed.

1. **Slow Learning (Outer Loop / "Evolution"):** The base parameters of the Dynamic State Propagation (DSP) blocks and the Neuromodulatory Controller (NMC) are learned slowly via standard backpropagation through time. This is the main training phase.
2. **Fast Adaptation (Inner Loop / "Lifetime Learning"):** The hidden state of each DSP block is treated as a set of "fast weights" that are updated at every time step via a learned, explicit delta rule. This is in-context learning.
3. **Global Control (Real-time Modulation / "Cognitive Control"):** The NMC provides top-down modulation of the fast adaptation process at every time step, adjusting the network's computational properties based on the global context.

This hierarchical learning structure is a direct implementation of the meta-learning ("learning to learn") principle. The "slow" outer loop learns a general-purpose, dynamic learning algorithm (the combination of the delta rule and neuromodulation), which is then executed "on the fly" by the "fast" inner loop during inference.

4.2 The "Mesa-Optimization" Framework

The entire system is best understood through the lens of **mesa-optimization**, as proposed in **"Transformers Learn In-Context by Gradient Descent"** (von Oswald et al.).

- **Objective:** The slow learning process does not just minimize a prediction error. It implicitly optimizes the parameters of the DSP blocks and the NMC to make them a better *inner learning algorithm*. The training objective is to produce a system whose forward pass is an efficient and effective optimization process for solving the task presented in the context.
- **The Learned Inner Loop:** The combination of the DSP's confidence-guided delta rule and the NMC's global modulation *is* the learned algorithm. The DSP's update rule, `h_t = h_{t-1} + β_t * (V_t - h_{t-1} * K_t)`, is a form of error-correcting update analogous to a gradient descent step. The NMC learns to be the "learning rate scheduler" for this process, adapting `β_t` and other parameters based on the context. This is precisely the kind of system described in the **"Self-Referential Weight Matrix"** (Irie, Schmidhuber et al.),

which learns to apply update rules to itself. The **TTT3R** paper (Chen et al.) provides a concrete example of this in the vision domain, framing it as "Test-Time Training."

4.3 Grounding in Statistical Physics and Emergent Structure

While we are designing the micro-level dynamics (the update rules), the macro-level behavior of the system should be understood through the lens of statistical physics, as detailed in **"Implicit Self-Regularization in Deep Neural Networks"** (Martin & Mahoney).

- **Heavy-Tailed Self-Regularization:** We hypothesize that the slow learning process will drive the "slow weights" of the N-DSM (the projection matrices in the DSP blocks and the weights of the NMC) into a "Heavy-Tailed" state. As the Martin & Mahoney paper argues, this is an emergent property of strongly correlated, well-optimized systems. This Heavy-Tailed distribution is not a bug; it is a feature, indicating that the network has learned to allocate its capacity across all scales, a hallmark of complex systems. The analogy to the **crystallization of glass** (Erlebach et al.) is apt: the optimization process "cools" the system from a random initial state into a highly structured, though not perfectly crystalline, final state.
- **Predictive Power of RMT:** This theoretical foundation provides us with powerful diagnostic tools. By analyzing the Empirical Spectral Density (ESD) of the weight matrices during and after training, we can quantitatively measure the amount of "implicit regularization" and correlation the model has learned. We can use this to compare different versions of the N-DSM and understand how architectural choices affect this emergent structure.

4.4 Connections to a Biological Theory of Mind

Finally, the N-DSM architecture provides a computable model for several high-level theories of brain function.

- **Predictive Coding and Free Energy Principle:** The interaction between the DSP blocks and the NMC can be seen as an implementation of predictive coding, a central theme in the **"Recurrent dynamics in the cerebral cortex"** paper (Singer). The internal state `h_t` of the DSP network represents the model's current "prediction" or "hypothesis" about the world. The input `x_t` is the sensory evidence. The NMC, by observing the discrepancy (or match) between the state and the input, learns to modulate the network to minimize future "prediction error." This entire process can be framed as the system learning to minimize its free energy, a foundational concept in computational neuroscience.
- **Complementary Learning Systems (CLS):** The **MemMamba** paper (Wang et al.) already hints at this. Our proposed architecture could be extended to more explicitly model CLS theory. The fast-adapting state of the DSP blocks acts like the hippocampus, rapidly

encoding episodic information. A separate, slower update mechanism (perhaps another NMC operating on a longer timescale) could be responsible for consolidating this information into the "slow weights" of the DSP blocks, analogous to the neocortex.

4.5 Summary

The task is to build a research paper and prototype for the **Neuromodulatory-Dynamic State Machine (N-DSM)**. This is a novel, brain-inspired LLM architecture designed to overcome the state and efficiency limitations of the Transformer.

- **The core component is the DSP Block:** A hybrid of a Mamba-like SSM for temporal state and MUDDFormer-like connections for dynamic hierarchical routing.
- **The state update is a learned delta rule:** It functions as a single step of an inner-loop optimization process, guided by a locally-computed confidence signal.
- **The global controller is the NMC:** A small, parallel recurrent network that learns to output neuromodulatory signals, which dynamically tune the operational parameters of all DSP blocks based on the global computational state.
- **The learning paradigm is meta-learning:** The system is trained via backpropagation to become a better and more efficient in-context learner.
- **The theoretical underpinnings** are drawn from State Space Models, the theory of Fast Weight Programmers, mesa-optimization, and the statistical physics of deep learning.

The initial goal is to create a PyTorch implementation of a single DSP block and then a full N-DSM, and to design experiments that can validate its performance on tasks requiring long-range state tracking and dynamic adaptation, directly addressing the failures identified in the Working Memory and Long-Term Conversational Memory benchmarks.

Acknowledgments

The author acknowledges the significant contribution of Google's Gemini 2.5 Pro in generating the initial concept for Neuromodulatory-Dynamic State Machine (N-DSM) and drafting the manuscript text. The author directed the generation process, curated the outputs, and takes full responsibility for the final content of this paper.