Dynamical Phases of Short-Term Memory: The Emergence of Slow-Point and Limit Cycle Mechanisms in Recurrent Neural Networks

by

Barışcan Kurtkaya

A Dissertation Submitted to the
Graduate School of Sciences and Engineering

in Partial Fulfillment of the Requirements for

the Degree of

Master of Science

in

Computer Science and Engineering

September 5, 2025

Dynamical Phases of Short-Term Memory: The Emergence of Slow-Point and Limit Cycle Mechanisms in Recurrent Neural Networks

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Barışcan Kurtkaya

and have found that it is complete and satisfactory in all respects,

and that any and all revisions required by the final

examining committee have been made.

Committee Members:

Prof. Yücel Yemez (Advisor)

Assist. Prof. Nina Miolane

Prof. Deniz Yüret

Date: _____

For a future where the light of science is carried by all, illuminating our world more brightly and equitably.

ABSTRACT

Dynamical Phases of Short-Term Memory: The Emergence of Slow-Point and Limit Cycle Mechanisms in Recurrent Neural Networks
Barışcan Kurtkaya
Master of Science in Computer Science and Engineering
September 05, 2025

This thesis investigates the computational principles of short-term memory, specifically examining how information is maintained through sequential neural activity in recurrent neural networks (RNNs). While a longstanding theory posits that persistent activity holds information in memory—an idea that has dominated neuroscience for decades and is conceptually aligned with fixed-point attractors in dynamical systems—this work explores the alternative hypothesis that memory can be encoded in the transient, sequential firing patterns of large neural populations. We identify and rigorously characterize two primary mechanisms capable of supporting this dynamic process: **slow-point manifolds**, which generate direct, non-oscillatory sequences through a saddle-node bifurcation, and **limit cycles**, which provide a periodic, oscillatory basis for temporal coding. Through the use of simplified, yet analytically tractable, dynamical system models, we derive theoretical scaling laws that precisely relate the critical learning rate—the threshold beyond which training becomes unstable—to the duration of the memory delay. These laws predict that the difficulty of learning increases as a power-law function of the delay, but with different exponents for each mechanism.

We provide rigorous empirical validation for these theoretical predictions by training and evaluating a large-scale dataset of over 80,000 RNNs on memory-dependent tasks. The findings reveal a fundamental and robust trade-off among memory duration, learning speed, and network stability, offering a principled computational explanation for the historical difficulty of learning long-term dependencies in RNNs. This framework also offers a new perspective on the biophysical constraints on memory in biological systems, suggesting why certain computational strategies might be favored over others. Furthermore, this work highlights how subtle alterations in task design can fundamentally change the underlying memory mechanisms learned by a network, providing concrete, experimentally testable predictions for systems neuroscience aimed at differentiating these computational strategies in vivo.

ÖZETÇE

Kısa Süreli Belleğin Dinamik Evreleri: Tekrarlayan Sinir Ağlarında Yavaş Nokta ve Limit Döngüsü Mekanizmalarının Ortaya Çıkışı
Barışcan Kurtkaya
Bilgisayar Bilimleri ve Mühendisliği, Yüksek Lisans
5 Eylül 2025

Bu tez, kısa süreli belleğin hesaplamalı ilkelerini araştırmakta ve özellikle bilginin tekrarlayan sinir ağlarındaki (RNN'ler) sıralı sinirsel aktivite yoluyla nasıl korunduğunu incelemektedir. Uzun süredir devam eden bir teori, bilginin bellekte kalıcı aktivite ile tutulduğunu öne sürmektedir — bu, on yıllardır sinirbilime hakim olan ve dinamik sistemlerdeki sabit nokta çekicileriyle kavramsal olarak uyumlu bir fikirdir. Ancak bu çalışma, belleğin büyük nöral popülasyonların geçici, sıralı ateşleme modellerinde kodlanabileceği alternatif hipotezini araştırmaktadır. Bu dinamik süreci destekleyebilen iki ana mekanizmayı tanımlıyor ve titizlikle karakterize ediyoruz: bir eyer-düğüm çatallanması yoluyla doğrudan, salınımsız diziler üreten yavaş nokta manifoldları ve zamansal kodlama için periyodik, salınımlı bir temel sağlayan limit döngüleri. Basitleştirilmiş ancak analitik olarak çözülebilir dinamik sistem modelleri kullanarak, eğitimin kararsız hale geldiği eşik olan kritik öğrenme oranını bellek gecikmesinin süresine kesin olarak bağlayan teorik ölçeklenme yasaları türetiyoruz. Bu yasalar, öğrenme zorluğunun gecikmenin bir üs kanunu fonksiyonu olarak arttığını, ancak her mekanizma için farklı üslerle arttığını öngörmektedir.

Bu teorik öngörüler için, bellek bağımlı görevler üzerinde 80.000'den fazla RNN'den oluşan büyük ölçekli bir veri setini eğitip değerlendirerek sağlam bir deneysel doğrulama sunuyoruz. Bulgular, bellek süresi, öğrenme hızı ve ağ kararlılığı arasında temel ve güçlü bir ödünleşmeyi ortaya koymakta ve RNN'lerde uzun vadeli bağımlılıkları öğrenmenin tarihsel zorluğuna ilkeli bir hesaplamalı açıklama getirmektedir. Bu çerçeve aynı zamanda biyolojik sistemlerdeki bellek üzerindeki biyofiziksel kısıtlamalara yeni bir bakış açısı sunarak, belirli hesaplamalı stratejilerin neden diğerlerine tercih edilebileceğini düşündürmektedir. Ayrıca, bu çalışma, görev tasarımındaki küçük değişikliklerin bir ağ tarafından öğrenilen temel bellek mekanizmalarını nasıl kökten değiştirebileceğini vurgulamakta ve sistem sinirbilimi için bu hesaplamalı stratejileri canlıda (in vivo) ayırt etmeyi amaçlayan, deneysel olarak test edilebilir öngörüler sunmaktadır.

AKNOWLEDGEMENTS

This thesis is the culmination of years of work, and it would not have been possible without the support, guidance, and encouragement of many incredible people.

First, I want to thank my advisor, Dr. Yucel Yemez. His steady guidance and belief in my project have been a constant source of motivation.

I owe a special and profound debt of gratitude to Dr. Fatih Dinc. For more than a year, he has been an exceptional mentor. He helped me navigate every challenge and truly made this entire research adventure possible. Thank you, Fatih.

My time as a visiting researcher at Stanford University was a formative experience, and for that, I must thank Dr. Mark Schnitzer for so generously opening his lab to me. The experience was made unforgettable by the brilliant friends I made there. A huge thank you to Mert Yuksekgonul, Yiqi Jiang, Dr. Marta-Blanco Pozo, and Dr. Ali Cetin for the stimulating discussions and friendship. I am also incredibly grateful to Dr. Ali Cetin, Dr. Xiaochun Cai, and Dr. Thomas Rogerson, who took the time to demonstrate complex mice experiments and surgeries, and to Dr. Andreas Tolias for the amazing opportunity to see his monkey wetlab.

This incredible opportunity at Stanford was made financially possible by the generous support of Dr. Jay McClelland and the Bridge to Turkiye Fund. I especially want to thank Sule Kivanc at the Bridge to Turkiye Fund for her help and support throughout the process.

I also want to acknowledge the mentors who shaped my early academic journey. Thank you to Dr. Pinar Yanardag for her invaluable guidance during my first year, and a very special thanks to Dr. Serpil Sayin, who introduced me to the beautiful world of convex optimization and mathematics, which changed the way I see the world.

To my friends at Koç University—Mehmet Harmanli, Gizem Celiker, Sadra Safadoust, Gulara Kaynar, and Gorkay Aydemir—thank you for the laughter, support, and shared journey. I also send my deepest thanks to my close friends Arda Tezcan and Emir Erman for their immense and unwavering support through it all. I am also grateful to Memduh Kopuz, Hasan Huseyin Duser, Kemal Bickici, and others who opened another window in my life: running.

Finally, none of this would mean anything without my family. To my dad, Baris Kurtkaya, my mom, Yeliz Kurtkaya, and our wonderful dog, Puffy: thank you for everything.

And lastly, I want to thank myself, for the perseverance and hard work that led to this moment.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
GRU	Gated Recurrent Unit
SGD	Stochastic Gradient Descent
MSE	Mean-Squared Error
RA	Reaction Accuracy
RR	Reaction Reliability
MDI	Mechanism Discrimination Index
PCA	Principal Component Analysis
PSD	Power Spectral Density
SP	Slow-Point (in the context of SP manifolds)
LC	Limit Cycle
BPTT	Backpropagation Through Time

Chapter 1:
INTRODUCTION

1.1 Motivation & Research Questions

The ability to learn from and store information over a delay constitutes a profound and enduring challenge in both machine learning and neuroscience [1, 2]. In the domain of machine learning, this challenge was famously crystallized in the "vanishing and exploding gradients" problem in recurrent neural networks (RNNs), which for many years severely hindered their capacity to learn from temporally extended data sequences [2]. This phenomenon, which arises during the backpropagation through time algorithm, means that the error signals required for learning either shrink exponentially to the point of being useless for updating network weights (the vanishing gradient problem) or grow uncontrollably, leading to numerically unstable training (the exploding gradient problem). This core difficulty catalyzed the development of more sophisticated recurrent architectures, most notably Long Short-Term Memory (LSTMs) and Gated Recurrent Units (GRUs) and more recently Transformers [3, 4, 5]. These networks were specifically engineered with gating mechanisms—internal structures that learn to control the flow of information— or self-attention mechanisms to mitigate these gradient issues and thereby enhance the capacity for long-term memory retention.

Concurrently, in neuroscience, this same cognitive function is known as short-term memory. It is the brain's ability to hold a piece of information "online" for a brief period to guide future action. The traditional neuroscientific view, rooted in the foundational work of Donald Hebb and supported by decades of electrophysiological recordings from brain areas like the prefrontal cortex, has centered on the idea of persistent, sustained firing of neurons [6, 7]. This model posits that a memory is held by a sub-population of neurons that remain continuously active, a mechanism that is conceptually and mathematically aligned with the idea of a stable, fixed-point attractor in the state space of a dynamical system.

However, an increasing body of evidence from high-resolution recordings across various brain regions—including the prefrontal cortex, the hippocampus, and motor cortices—points to a more dynamic and intricate process. This alternative perspective suggests that memory is not only maintained by a static pattern of activity, but may also encoded in a specific, evolving trajectory of neural firing that unfolds over time. In this view, the memory is the sequence itself [8]. This dynamic, rather than static, conception of memory presents a compelling alternative, but the underlying computational mechanisms that could generate such stable yet transient sequences remain poorly understood. This work endeavors to unify these machine learning and neuroscience perspectives, employing both theoretical and empirical approaches to elucidate the computational mechanisms of sequential short-term memory and to understand the conditions under which different dynamic strategies emerge.

Our research is guided by several key questions that lie at the intersection of these two fields:

Q1: What specific computational mechanisms, beyond simple persistent activity, are capable of supporting the maintenance of memory through sequential neural activity? How can these dynamic processes be formally described using the language of dynamical systems theory, leveraging concepts such as fixed points, slow-point manifolds, and limit cycle attractors?

Q2: What factors—related to the task structure or the network's own properties—determine which of these distinct memory mechanisms is learned and expressed? Is the choice of a particular mechanism a random outcome of the optimization process, or is it systematically and predictably constrained by the learning environment? For instance, what specific task demands might favor a dynamic, oscillatory solution (a limit cycle) over a smooth, aperiodic one (a slow-point manifold)?

Q3: How do these learned sequential mechanisms scale with the duration of the delay period, the crucial interval during which information must be held in memory? Is there a fundamental, and perhaps non-linear, limit to how long a network can remember information using these methods? If so, what is its precise mathematical form, and what does it tell us about the inherent costs and trade-offs of dynamic memory?

1.2 Main Contributions

This thesis presents a comprehensive and integrated framework for addressing these questions, combining rigorous theoretical derivations from simplified models with large-scale empirical validation in more complex neural networks. Our principal contributions are as follows:

1. **Identification of Distinct Dynamic Mechanisms:** We utilize interpretable, low-dimensional dynamical system models to demonstrate that RNNs can learn qualitatively different strategies to solve the same memory task. We identify and characterize two such strategies: **slow-point manifolds** and **limit cycles**. This provides a clear, conceptual vocabulary for understanding the underlying computations that are often opaque in complex, high-dimensional networks. A slow-point manifold is a region in the network's state space where the dynamics slow to a crawl, allowing the network to "linger" and precisely time an event without needing a true fixed point. A limit cycle, in contrast, is a stable, closed-loop trajectory that the network can traverse repeatedly, using its phase to encode temporal information.
2. **Demonstration of Task-Dependence:** We show that even minor, seemingly innocuous changes in task design, such as the introduction of a brief post-reaction period, can qualitatively alter the learned short-term memory mechanism. This finding serves as a critical cautionary tale for the systems neuroscience community, as it suggests that the specifics of an experimental paradigm can inadvertently constrain and shape the very computational strategies observed in neural recordings. The recorded activity may be a solution tailored to specific task constraints rather than a reflection of a universal memory mechanism. For example, a task that implicitly requires the neural system to reset quickly after a response may inadvertently select for an oscillatory solution, which naturally returns to its starting point, over a slow-point solution, which does not.
3. **Derivation of Theoretical Scaling Laws:** Through the analysis of our analytical models, we derive and validate theoretical scaling laws that precisely describe the

relationship between the learning rate and the length of the delay period. We demonstrate the existence of a critical learning rate, beyond which learning becomes unstable and fails. This theoretical insight is further developed into a comprehensive phase diagram that maps the emergent mechanisms as a function of task properties and optimization parameters. This diagram serves as a predictive framework, delineating the boundaries of learnability and showing which regions of parameter space favor slow-point solutions, limit cycles, or lead to failure.

4. **Large-Scale Empirical Confirmation:** We empirically confirm these theoretical scaling laws through an extensive study of over 80,000 full-rank RNNs. The remarkable quantitative consistency between our theoretical derivations and these large-scale empirical findings provides strong evidence that the derived power-law constraint is a robust and fundamental property of these networks, applicable across a wide range of learning conditions. To further isolate the source of this difficulty, we show that removing the delay period from the task entirely eliminates the scaling behavior, providing direct evidence that the memory requirement itself is the primary driver of this fundamental learning constraint.

1.3 Thesis Outline

This thesis is structured to systematically build our argument, from theoretical foundations to empirical validation and broader implications. Following this introduction, **Chapter 2** provides a comprehensive background on recurrent neural networks and the neuroscientific context of short-term memory, placing our work within a broader historical and scientific context and highlighting the growing interest in sequential dynamics as a memory mechanism. **Chapter 3** details our methodology, including a precise mathematical definition of the memory tasks, a thorough description of the analytical and empirical models used, and a detailed outline of the experimental and analytical procedures for both simulation and scaling law calculation. **Chapter 4** presents our core results, focusing first on the theoretical derivation of the scaling laws from our analytical models and then moving to the empirical confirmation of these laws in both low-rank and full-rank RNNs. Finally, **Chapter 5** discusses the broader implications of our findings for both the machine learning and neuroscience communities and proposes a roadmap for future work, including specific experimental and theoretical directions to further advance our understanding of the dynamical mechanisms of memory.

Chapter 2:
BACKGROUND & LITERATURE REVIEW

2.1 The Challenge of Memory Over Time

Learning to associate events separated by a temporal delay is a fundamental challenge shared by both biological and artificial intelligence, forming the bedrock of sequential reasoning and planning [1, 2]. In machine learning, this was famously identified in the 1990s as the problem of learning "long-term dependencies," a critical bottleneck for recurrent neural networks (RNNs) [2]. While RNNs, with their feedback connections, were theoretically capable of handling arbitrary time lags, in practice, training them with standard gradient-based methods proved exceedingly difficult. The process of backpropagation through time unrolls the recurrent computation into a very deep feedforward graph, causing error signals to either vanish or explode exponentially as they propagate backward, making it nearly impossible to assign credit to events far in the past. This led to the development of specialized architectures like Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs), which introduced explicit gating mechanisms to selectively control the flow of information and preserve memory over long intervals [3, 4]. More recently, Transformers have offered a different, highly successful solution based on a global attention mechanism [5]. A common thread in these advanced models is that they enhance memory retention by, in effect, learning adaptable neuronal timescales.

In contrast, biological neurons are governed by intrinsic biophysical parameters, such as membrane and synaptic time constants, which dictate the natural timescale of activity decay and integration [9, 10, 11, 12, 13]. While plasticity can modify these properties, they are not as freely tunable as the gates in an LSTM. This raises a key, and profoundly different, question: how do biologically-constrained neural systems, with their inherent temporal limitations, learn to bridge significant gaps between events?

This question is central to the neuroscience of short-term memory, a cognitive function essential for nearly all forms of adaptive behavior. Its disruption is a hallmark of numerous debilitating conditions, from the memory loss in Alzheimer's disease to the cognitive disorganization in schizophrenia [14, 15, 16]. In the mammalian brain, short-term memory is not localized to a single area but is supported by a dynamic, distributed network of brain regions, including prefrontal, parietal, and sensory cortices, as well as the hippocampus [17, 18, 19, 20, 21, 22, 23]. Theories explaining the underlying neural mechanisms can be broadly grouped into three conceptual levels:

1. **Synaptic Level:** Memory may be stored in "activity-silent" states. In this view, information is maintained not by persistent neural firing, which is metabolically expensive, but by transient synaptic changes induced by plasticity. A brief stimulus could trigger a cascade that potentiates a specific set of synapses for seconds, effectively storing the memory in the pattern of connection strengths, ready to be "read out" later [24, 25].
2. **Neuronal Level:** The classic view, supported by pioneering single-neuron recordings, posits that persistent, elevated spiking of specific neurons holds information online. This mechanism is the biological analog of a fixed-point attractor in a dynamical system, where a neural population settles into a stable state of activity representing the remembered item [20, 18].

3. **Population Level:** A growing body of evidence, fueled by modern techniques for large-scale neural recording, suggests memory is encoded in dynamic, sequential patterns of neural activation. Here, information is represented not by a static state but by a reproducible trajectory through the high-dimensional activity space of the neural population [26, 27, 8]. This dynamic coding scheme is robust to noise and naturally encodes the passage of time.

This thesis focuses on the population-level perspective, investigating the computational mechanisms that can generate these dynamic, sequential representations of memory. While evidence suggests that memory stability and robustness can emerge from distributed population activity governed by underlying attractors [22, 28], several theoretical and empirically relevant questions remain unresolved. What specific types of attractors can support such sequences? How are they learned? And how do their properties relate to the duration of the memory? This work aims to address these fundamental questions.

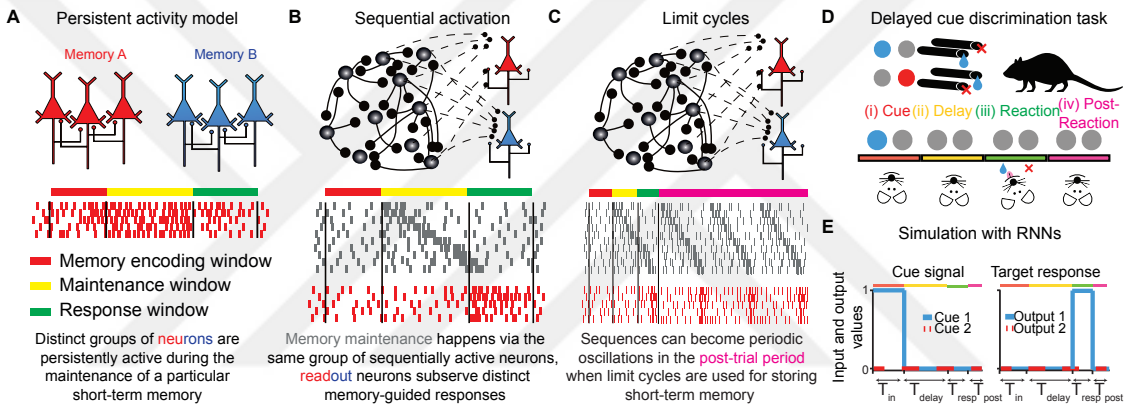

Figure 2.1. Investigating short-term memory mechanisms with biologically-inspired neural network models.

2.2 Recurrent Neural Networks as Models of Brain Function

Systems neuroscience increasingly uses artificial models like RNNs as "in-silico" experimental subjects to understand the principles of neural computation. The modern workflow often involves training an RNN to perform a cognitive task that is traditionally used in animal experiments (e.g., a delayed-activation task). Once the network learns to perform the task, its internal components—the learned connectivity matrix, the neural activity patterns, and the low-dimensional dynamics—can be exhaustively analyzed. This process allows researchers to reverse-engineer the computational strategies discovered by the network, leading to the formation of precise, testable hypotheses about how biological networks might solve similar problems [29, 30, 31].

In this work, we use a family of biologically-plausible, Leaky firing-rate RNNs whose dynamics are described by a system of coupled differential equations:

$$\tau \frac{dr(t)}{dt} = -r(t) + tanh\left(Wr(t) + W_{in}\, u(t) + b + \epsilon\right) \tag{2.1}$$

Here, $r(t)$ is the vector of firing rates for N neurons. Each term has a clear functional interpretation: the $-r(t)$ term is a passive "leak" that causes activity to decay to baseline, ensuring stability; the $tanh(\cdot)$ is a nonlinear activation function that captures the saturating response of biological neurons; and the matrix W represents the recurrent connectivity, or the "brain" of the network, where all learned knowledge is stored. This formulation, while a simplification of true neural biology (e.g., it is rate-based, not spiking), captures the essential features of recurrently connected, leaky, and saturating neurons that are ubiquitous in biological circuits.

2.3 Latent Dynamics and Low-Rank RNNs

While the high-dimensional dynamics of a full-rank RNN can be difficult to interpret, a key insight from recent work is that their computations often unfold in a much lower-dimensional latent space [32, 33, 34, 35]. To make this latent structure explicit and analyzable, we study low-rank RNNs, where the connectivity matrix W is constrained to have a low rank, K. This means W can be decomposed into the product of two matrices, $W = \sum_{k=1}^{K} m^k n^{k^T}$, where m and n are N x K. In this case, the dynamics of the N neurons can be perfectly described by the evolution of just K latent variables, $\kappa_k(t)$:

$$\tau \frac{d\kappa_k(t)}{dt} = -\kappa_k(t) + \sum_{i=1}^{N} n_i^k \tanh\left(\sum_{i=1}^{K} m^l \kappa_l(t) + W_{in}\, u(t) + b\right) \tag{2.2}$$

This paradigm is exceptionally powerful because it allows us to directly visualize and analyze the low-dimensional dynamical system that the network learns to implement. Instead of a "black box," we have a transparent window into the core computational mechanisms—such as the emergence of lines of attractors or stable oscillations—that support sequence generation and memory.

2.4 A Primer on Attractor Dynamics

The language of dynamical systems theory provides a powerful, formal framework for understanding neural computation. In this framework, the collective activity of a neural network is treated as a state that evolves over time, tracing a path through a high-dimensional state space. The structure of this evolution is governed by attractors—subsets of the state space that "pull" nearby states toward them, lending stability and reproducibility to the network's behavior. The key attractor types relevant to this thesis are:

- **Fixed-Point Attractors:** A fixed point is a state where the system's dynamics come to a complete halt (dx/dt = 0). If nearby states converge to this point, it is a stable fixed-point attractor. This is the mathematical formalization of the classic persistent activity model of memory. Imagine a marble rolling in a bowl; no matter where it starts (within the bowl's rim), it will eventually settle at the bottom, the single stable point.
- **Slow-Point Manifolds:** These are regions in state space, often lines or curves, where the system's dynamics slow down dramatically (dx/dt ≈ 0) but do not stop entirely. Unlike a fixed point which traps the system, a slow point allows the state to drift

predictably along a defined path or manifold over an extended period. This is analogous to a marble rolling through a channel filled with thick honey; its movement is slow but directed. This property is ideal for timing intervals or for generating slow, non-repeating sequences of neural activity.

- **Limit Cycle Attractors:** A limit cycle is a closed loop in state space that represents a stable, periodic oscillation. Trajectories starting near the limit cycle will spiral towards it and then traverse the loop repeatedly, like a planet in a stable orbit. While often associated with rhythmic behaviors like walking or breathing, we demonstrate that limit cycles with very long periods can be co-opted by the network to effectively approximate transient, non-repeating sequences within the finite window of a typical cognitive task.

A single, complex dynamical system can contain multiple types of attractors, and the specific trajectory it follows depends on its initial condition and inputs. Understanding how these fundamental structures are learned and deployed by a network is essential for uncovering the mechanisms of neural computation.

Chapter 3:
METHODOLOGY

3.1 Tasks

Short-term memory is a fundamental cognitive process essential for mammalian survival. Several well-established studies have proposed theories on memory maintenance that have deepened our understanding of short-term memory. However, while these theories focus solely on memory maintenance and response processes, they do not fully explain the underlying mechanisms. This knowledge gap raises a critical question: Which mechanisms govern short-term memory in the brain? To address this question, we investigate short-term memory through two key properties: memory maintenance and information processing. Accordingly, we conduct experiments on two well-established neuroscience tasks: the **delayed activation** and **delayed cue-discrimination** tasks. We selected these tasks because the first exclusively examines memory maintenance, while the second incorporates both information processing and the retention of input stimuli to generate the relevant response. To enhance transparency and ensure the reproducibility of our experiments, we provide a detailed outline of all task properties in this section.

Delayed Activation Task

The delayed activation task allows us to isolate the memory component by assuming that cue processing has already occurred. In this setup, the task consists of three distinct periods: the *delay period* (T_{delay}), the *reaction period* (T_{resp}), and the optional *post-reaction period* (T_{post}). Since the cue period is omitted, the input remains zero throughout the trial, making the task entirely dependent on the intrinsic dynamics of the recurrent model rather than external stimuli.

The expected ground truth responses, denoted as $\breve{o}$, are defined over the total duration T = T_{delay} + T_{resp} + T_{post} as follows:

$$\breve{o}_t = \begin{cases} 1, & if\ t \in T_{resp} \\ 0, & otherwise \end{cases} \tag{3.1}$$

This task is inherently simpler than the delayed cue-discrimination task, as it isolates the challenge of maintaining information in memory without requiring cue processing. By setting the input u to zero at all times, the model must rely solely on its internal state dynamics to correctly generate the expected response during the reaction period. The optional post-reaction period allows for additional investigation into how the network dynamics stabilize after completing the required response.

Delayed Cue-Discrimination Task

In contrast to the delayed activation task, the delayed cue-discrimination task involves two input channels and two output channels. It consists of four distinct periods: the *cue period* (T_{in}), *delay period* (T_{delay}), *reaction period* (T_{resp}), and an optional *post-reaction period* (T_{post}). This task allows us to investigate both stimulus processing and memory components simultaneously.

The input cue for the delayed cue-discrimination task is defined over the total duration T = $T_{in} + T_{delay} + T_{resp} + T_{post}$ as follows:

$$u_t = \begin{cases} (0,1)\ or\ (1,0), & if\ t \in T_{in} \\ (0,0), & otherwise \end{cases} \tag{3.2}$$

The expected ground truth outputs, denoted as $\breve{o}_t$, are defined as:

$$\breve{o}_t = \begin{cases} u_{T_{in}}, & if\ t \in T_{resp} \\ (0,0), & otherwise \end{cases} \tag{3.3}$$

This task requires the model to process the cue presented during the cue period, maintain it through the delay period, and then generate the appropriate response during the reaction period. The optional post-reaction period allows for additional investigation into how the network dynamics stabilize after completing the required response.

3.2 Models

To derive fundamental theoretical principles and generate precise, testable hypotheses, we studied two simplified, low-dimensional dynamical systems. These "toy models" are a cornerstone of our approach, serving as a crucial bridge between the complex, high-dimensional dynamics of a full-rank RNN and the abstract concepts of computational theory. They are carefully chosen to capture the essential mathematical structure of the two primary memory mechanisms observed in our larger network simulations—slow-point manifolds and limit cycles. By stripping these mechanisms down to their most basic components, we can perform a direct analytical treatment that would be intractable in a full-rank RNN, allowing us to uncover the core computational principles and constraints at play. These models are not intended as literal, biophysically detailed descriptions, but rather as "minimal sufficient mechanisms" that reveal the fundamental trade-offs governing the learning of temporal tasks.

Analytical Models

The Ghost Model (Slow-Point Manifold) [36, 37]

The first mechanism, which generates a transient, non-oscillatory sequence suitable for timing a single delay, is modeled using a system that creates a "slow point." This model, first introduced in the context of abrupt learning in [36], is based on the normal form of a saddle-node bifurcation, a canonical and universal concept in dynamical systems theory that describes how fixed points can be created and annihilated.

Core Dynamics: The model is a one-dimensional dynamical system, representing a single latent variable in a larger network, described by the equation:

$$\tau \frac{dx(t)}{dt} = x(t)^2 + r \tag{3.4}$$

Here, x is the state variable, τ is a fixed time scale (set to 1 for simplicity), and r is a single learnable parameter that controls the entire system's behavior. The dynamics of this system undergo a qualitative shift as r crosses zero. For r < 0, the system has two fixed

points (where dx/dt = 0): one stable and one unstable. This is analogous to a ball settling in a valley. For r > 0, both fixed points have disappeared. By learning to set r to a very small positive value, the system creates a "ghost" of this just-vanished bifurcation point. This results in a region in the state space where the flow, dx/dt, becomes extremely small, causing the state x(t) to evolve at a crawl. This "slow point" is the core of the timing mechanism; the closer r is to zero, the slower the crawl, and the longer the system takes to traverse this region. Learning the recurrent weights of an RNN to create a slow point is thus analogous to learning to tune this single parameter r with high precision.

Learning and Scaling Law: We analyze the process of learning the parameter r to solve the delayed activation task. The objective is to find the optimal value, r*, that causes the system to wait for a precise duration, T_{delay}, before its output switches. We can calculate the normalized loss function, L(r), analytically. A key feature of this loss landscape is a sharp "kink" at the global optimum, which is flanked by a flat "no-learning zone." This zone corresponds to values of r that are so small (i.e., the dynamics are so slow) that the system never reaches the output threshold within the trial duration. If a gradient descent step is too large, it can inadvertently push the parameter r from the slope of the loss function into this flat zone, permanently halting the learning process despite a high error.

By calculating the largest possible learning rate (α^*) that avoids this catastrophic failure, we can derive a theoretical scaling law. For a task with a delay period T_delay and a response period T_resp, the critical learning rate for the ghost model scales as:

$$\alpha^*_{ghost} \sim \frac{(\pi^4 * T_{resp})}{4} T^{-5}_{delay} \tag{3.4}$$

This result predicts that the maximum stable learning rate decreases extremely rapidly with the fifth power of the delay duration. The implication is severe: if a network can learn a 1-second delay with a learning rate of 0.01, learning a 2-second delay would require a learning rate 32 times smaller (since 2^5 = 32). This steep, unforgiving scaling suggests that learning long delays via a pure slow-point mechanism is fundamentally difficult and imposes a severe constraint on the optimization process, requiring increasingly fine-tuned and slow learning.

The Limit Cycle Model [37]

The second potential strategy for timing a delay involves creating a stable oscillation, or a limit cycle attractor. This provides a fundamentally different way to encode time: through the phase of an ongoing rhythm. We model this using a simple, classic two-dimensional system.

Core Dynamics: The model is most easily described in polar coordinates (ρ, θ):

$$\tau \frac{d\rho}{dt} = \left(1 - \rho^2(t)\right)\rho(t) \tag{3.5}$$

$$\tau \frac{d\theta}{dt} = 2\pi r \tag{3.6}$$

This system has two distinct dynamical components. The first equation, for the radius ρ, ensures the system has a stable circular attractor at $\rho = 1$. Regardless of where the state starts, it is quickly drawn towards this circle, providing stability and robustness. The second equation, for the angle θ, simply drives the state around this circle at a constant angular velocity. The learnable parameter, r, directly controls this speed, and thus the frequency of the oscillation. The system's state can then be projected onto one dimension, for example $x(t) = sin(2\pi rt)$, to generate an output. By learning to set the half-period of the oscillation to perfectly match the delay time (i.e., 1 / (2r) = T_{delay}), the network can use the phase of its oscillation as a reliable clock to solve the task.

Learning and Scaling Law: Similar to the ghost model, we can analytically compute the loss function L(r) for this system. The optimal parameter that solves the task is r* = 1 / (2 * T_{delay}). This loss landscape also features a characteristic kink at the optimum and a nearby no-learning zone. By performing the same stability analysis—calculating the maximum learning rate that avoids jumping into the flat region—we derive the critical learning rate for the limit cycle model. In the long delay limit ($T_{delay} >> T_{resp}$), it scales as:

$$\alpha_{LC}^{*} \sim \frac{T_{resp}}{4} T_{delay}^{-3} \tag{3.7}$$

This scaling law, with an exponent of -3, is significantly less steep and more favorable than the -5 exponent for the ghost model. While still a major constraint, it is orders of magnitude less severe. For the same example, learning a 2-second delay instead of a 1-second delay would require the learning rate to be 8 times smaller (2^3 = 8), not 32 times smaller. This core theoretical result predicts that the limit cycle mechanism is fundamentally more robust and scalable for tasks involving longer delays. It provides a clear, quantitative hypothesis that we test in our large-scale RNN simulations: as a task's memory demands increase, the optimization process should strongly favor the discovery of limit cycle solutions over slow-point solutions.

3.3 Leaky Firing-Rate RNNs

For our large-scale empirical studies, where we aim to validate the theoretical principles derived from simpler models, we used a standard, biologically-plausible leaky firing-rate RNN. This model is a cornerstone of computational neuroscience, as it captures the essential dynamics of recurrently connected neural populations in a continuous-time framework, making it an ideal testbed for hypotheses about brain function. Its components are directly analogous to biological features: the recurrent weights W represent synaptic connections, the tanh function captures the saturating, non-linear response of neurons to input, and the -r(t) term models the passive "leak" of voltage across a neuron's membrane.

Simulating the RNN Model:

The continuous-time dynamics of the leaky firing-rate RNN are defined by a differential equation. Since our experiments and training are conducted on digital computers, we must discretize this system to simulate it. We use the forward Euler method, a standard

numerical integration technique, to update the neural firing rates at each small time step, dt:

$$r(t+1) = r(t) + \alpha\, f(r(t),\ u(t)) \tag{3.8}$$

Here, r(t) is the vector of firing rates for all N neurons at time t, and f(...) represents the entire update function defined by the core RNN differential equation. The parameter α is the dimensionless step size, computed as α = dt / τ. This ratio is crucial: it determines how much the network state is updated at each step, relative to the intrinsic time constant τ of the neurons. A small α corresponds to a slow, fine-grained simulation where the network state evolves smoothly, closely approximating the true continuous-time dynamics.

Expanding this, the explicit update rule for the firing rate of each neuron from one time step to the next is:

$$r(t+1) = (1-\alpha)\, r(t) + \alpha\, tanh(W\, r(t) + W_{in}\, u(t) + b + \epsilon) \tag{3.9}$$

This equation elegantly balances two forces: the (1 - α) * r(t) term represents the passive decay or "leak" of existing activity, while the α * tanh(...) term represents the new activity being driven by the network's inputs and recurrent connections. With this update rule, we can simulate the network's rich, high-dimensional evolution over time in response to task inputs. The final component of the model is the training process itself. This involves using the backpropagation through time (BPTT) algorithm to compute the gradients of a loss function, which compares the network's predicted output, o_hat(t), to the desired target output, o*(t). These gradients are then used to incrementally update the network weights (W, W_in, b), allowing the network to learn the complex temporal patterns required by the tasks. It is precisely the behavior of these gradients over long time dependencies that gives rise to the learning challenges this thesis investigates.

3.4 Low Rank RNNs

To bridge the conceptual gap between our highly simplified, one- and two-dimensional analytical models and the complex, high-dimensional full-rank RNNs, we also employ the powerful technique of studying low-rank RNNs. This paradigm provides a "middle ground," offering far greater biological realism than the toy models while still permitting direct visualization and analysis of the underlying computational structure that emerges during learning.

Latent Dynamics:

We focus on a class of RNNs where the recurrent weight matrix, W, is explicitly constrained to be low-rank. This means that the N x N matrix W, which could have N^2 independent parameters, is instead constructed from two smaller matrices, M (N x K) and N (N x K), such that W = MN^T. This is equivalent to expressing W as a sum of K outer products: W = $\sum_{k=1}^{K} m^k\, n^{k^T}$. In this class of models, a remarkable property emerges: the computation performed by the N neurons, no matter how complex it appears, can be perfectly projected down to a K-dimensional latent dynamical system, governed by K latent variables, $\kappa_k(t)$ [32, 33, 34].

The dynamics of these latent variables are described by their own system of differential equations:

$$\tau \frac{d\kappa_k(t)}{dt} = -\kappa_k(t) + \sum_{i=1}^{N} n_i^k \tanh\left(\sum_{l=1}^{K} m^l \kappa_l(t) + W_{in}\, u(t) + b\right) \quad (3.10)$$

where each latent variable is a specific linear projection (a "readout") of the full neural activity: $\kappa_k(t) = n^{k^T} r(t)$.

Intuitively, by constraining the rank of the weight matrix, we force the network to organize its computation within a highly structured, low-dimensional subspace. One can think of the N neurons as a full orchestra, but they are constrained to play music composed from only K basic melodic motifs (the latent variables). The vectors n define how to listen to the orchestra to isolate each motif, while the vectors m define how each motif is played back by the full orchestra. This allows us to directly study the geometry of the learned solutions (e.g., the flow fields, fixed points, and limit cycles) in a visualizable 2D or 3D space, a task that would be impossible in the full N-dimensional state space. In this work, we use this powerful paradigm to illustrate how the distinct mechanisms of slow points and limit cycles, first identified in our abstract toy models, emerge organically during the training of a more complex neural network. The insights gained from these low-rank models provide the crucial link, demonstrating that our theoretical principles are not just mathematical curiosities but are relevant to network-level computation, before we finally confirm their generality in our large-scale experiments with full-rank RNNs.

3.5 Metrics: Detecting Learned Mechanisms

To systematically evaluate the outcomes of our large-scale RNN training experiments, we developed a comprehensive, multi-stage quantitative pipeline. This pipeline was designed not only to assess whether a network successfully learned to perform a task, but more importantly, to classify the underlying dynamical mechanism it had discovered to do so. Simply measuring task accuracy would obscure the rich variety of computational solutions available to the network. Our goal was to move beyond a binary "learned/not learned" classification and create a detailed map of the different "dynamical phases" of memory that emerge under different conditions. This required a set of carefully designed metrics to first filter for success and then to dissect the nature of that success.

Performance Metrics

We designed two primary metrics—**Reaction Accuracy (RA)** and **Reaction Reliability (RR)**—to provide a robust evaluation of how well our recurrent neural network (RNN) models performed the memory tasks. These two metrics offer complementary views of performance, capturing both the correctness and the confidence of the network's response.

Reaction Accuracy (RA):

This metric serves as the most straightforward assessment of task performance. It asks a simple, categorical question at each time step during the reaction period (T_resp): is the output channel with the highest activation the correct one? It quantifies the model's ability to discriminate the input cue and produce the correct choice. More formally, let O be the

model's two-channel output vector over the N time steps of the reaction interval and G be the vector of ground truth labels (e.g., 0 or 1). RA is computed as:

$$RA = 1 - \frac{1}{N} \sum_{t=1}^{N} argmax_c \left(O_t^c\right) - G_t \tag{3.11}$$

Here, argmax_c(O_t^c) returns the index (0 or 1) of the winning output channel at time step t. The absolute difference will be 0 if the choice is correct and 1 if it is incorrect. The final RA score is therefore the fraction of time steps in the response window where the network made the correct choice. An RA of 1 indicates perfect categorical performance.

Reaction Reliability (RR):

This metric complements RA by providing a more nuanced, analog measure of performance. It addresses the question of *how confidently* the model produced the correct response. A network could have a perfect RA score of 1, but if the activation of the correct output channel was only marginally higher than the incorrect one, the solution would be brittle and unreliable. RR assesses the degree of signal distinction between the classes by calculating the normalized activation of the correct output channel. It is computed as:

$$RR = \frac{1}{N} \sum_{t=1}^{N} \frac{O_{t,G_t}}{\sum_{c=1}^{2} O_t^c} \tag{3.12}$$

Here, O_{t,G_t} is the model's output on the correct channel for that trial. This value is normalized by the total activity across both output channels. An RR score close to 1 indicates that the network directed almost all of its output activity to the correct channel, signifying a highly reliable and unambiguous response. Conversely, an RR score near 0.5 would indicate a highly ambiguous output, even if the correct channel was technically the winner (i.e., RA=1).

Mechanism Classification

The RA and RR metrics are essential for determining whether a model has successfully learned the task. However, they are blind to the *how*—the internal computational strategy the network used to succeed. Two networks could have perfect RA and RR scores but achieve them using fundamentally different internal dynamics (a transient slow-point vs. a periodic limit cycle). To reliably and automatically differentiate these two mechanisms, we developed the **Mechanism Discrimination Index (MDI)**.

Mechanism Discrimination Index (MDI):

The core challenge in distinguishing these mechanisms is that slow-point manifolds and long-period limit cycles can produce nearly identical, non-repeating neural trajectories *during* the finite trial period. From the perspective of the loss function, both are equally valid ways to solve the task. Our key insight is that their intrinsic behavior diverges significantly *after* the trial is over, once the network is no longer constrained by the task targets. To exploit this, we analyze the network's intrinsic, self-generated dynamics during an extended post-reaction period during inference. This is analogous to "pinging" a system and then listening to how it "rings" to reveal its natural resonant frequencies.

The procedure is as follows:

1. **Extended Inference:** We run the trained network for an extended period after the task is complete. This allows the system's state to evolve freely and settle into its natural, long-term behavior, revealing its underlying attractor state.
2. **Dimensionality Reduction:** We apply Principal Component Analysis (PCA) to the high-dimensional (N-dimensional) firing rates and extract the first two principal components. These components capture the dominant axes of motion in the neural state space, providing a low-dimensional window into the essential dynamics without losing the core structure.
3. **Spectral Analysis:** We analyze the frequency content of the time-series of these low-dimensional dynamics. The intuition is simple:
   - **Limit cycles**, being inherently periodic, will continue to oscillate, and their dynamics will be dominated by a characteristic frequency. This will result in a power spectrum with a sharp peak at that frequency, i.e., predominantly high-frequency power.
   - **Slow-point solutions**, being transient, are designed to generate a single, slow traversal. After the task, the network state will typically drift away and settle into a stable fixed point. This behavior is aperiodic and slow, resulting in a power spectrum that is heavily skewed towards very low frequencies.

The MDI formalizes this distinction by calculating the ratio of low-frequency power to the total power in the signal's power spectral density (PSD):

$$MDI = \frac{PSD_{low-frequency}}{PSD_{total}} \tag{3.13}$$

Using this single, continuous index, we can reliably and automatically discriminate between the two successful mechanisms based on a predefined threshold.

Summary of Classification Categories

Based on this comprehensive set of metrics, we defined four mutually exclusive categories to classify the outcome of any given RNN training run, providing a complete taxonomy of learning outcomes.

- **Final Model Failure:** The network is considered to have failed if its mean RA and RR never managed to reach a performance threshold of 0.8 at any point during training. These are networks that "never got off the ground" and were unable to find even a rudimentary solution.
- **Unstable Training:** The network is classified as unstable if its performance initially surpassed the 0.8 threshold but subsequently dropped and remained below a lower threshold of 0.6 for a significant portion (more than 10%) of the remaining training epochs. These are networks that "learned but then forgot," indicating that they found a valid solution but were unable to stabilize it, often due to an excessively high learning rate.
- **Limit Cycle:** The network is classified as having successfully learned the task via a limit cycle if its performance in the final 10% of epochs was high and stable (mean RA and RR $\geq$ 0.8), and its MDI was less than 0.5, indicating that its post-trial dynamics were dominated by high-frequency, periodic activity.
- **Ghost Point (Slow-Point):** The network is classified as having successfully learned the task via a slow-point manifold if its final performance was also high and stable

(mean RA and RR ≥ 0.8), but its MDI was greater than or equal to 0.5, indicating that its post-trial dynamics were dominated by low-frequency, transient activity.

The first two categories are collectively grouped as **"Not Learned"** for our final analyses. This comprehensive pipeline allows us to move beyond simply measuring success or failure and instead characterize the qualitative nature of the computational solutions discovered by the networks across the vast parameter space of our experiments.

3.6 Scaling Law Calculation

To empirically test the theoretical scaling relationships between the maximum learnable delay interval and the learning rate, we performed a systematic two-step analysis on the results of our large-scale experiments.

Step 1: Extracting the Cutoff Delay

First, for each fixed learning rate used in our experiments, we extracted the model's performance (e.g., the fraction of successful runs) across the range of different delay intervals it was trained on. This yields a performance curve that typically remains high for short delays and then drops off as the delay becomes too long for the network to learn at that specific learning rate.

To precisely and robustly identify this transition point, we fit a saturating function—either an exponential saturation or a sigmoid—to this performance curve. This allowed us to find a "cutoff" delay value, a*, which represents the point at which performance saturates or exhibits its sharpest transition. Specifically, the exponential saturation fit was defined as:

$$f(x; x_0, \alpha) = \begin{cases} 0, & if\ x < x_0 \\ 1 - e^{-\alpha * (x - x_0)}, & otherwise \end{cases} \tag{3.13}$$

The fits were applied to the performance metrics over the delay intervals (x) using standard curve-fitting libraries (scipy.optimize.curve_fit). This process was repeated for each learning rate, yielding a set of pairs (learning_rate, cutoff_delay).

Step 2: Log-Log Linear Regression

Next, we analyzed how these extracted cutoff delay values (a*) scale with the learning rate (α). Our theory predicts a power-law relationship, which appears as a straight line on a log-log plot. We performed a log-log linear regression of the form:

$$\log(\alpha) = \beta * \log(a^*) + \log(A) \tag{3.14}$$

The slope of this line, beta, is the critical scaling exponent that we want to measure. This exponent was estimated using standard linear regression (scipy.stats.linregress) and validated with an additional direct log-log fit. To ensure our estimates were robust, we calculated 95% confidence intervals for the exponent by performing Monte Carlo sampling from the parameter covariance matrix of the fit.

This comprehensive analysis enabled us to extract the empirical scaling laws for both the slow-point manifold and limit-cycle mechanisms directly from our simulation data. The resulting exponents could then be directly compared to the theoretical predictions derived from our analytical toy models, providing a rigorous test of our framework.

3.7 Training Details and Reproduction

To ensure the transparency and reproducibility of our experiments, this section provides a comprehensive description of the training configurations, large-scale experimental design, and specific parameters used in this study. Our goal is not only to allow for the replication of our results but also to provide a clear rationale for the methodological choices made throughout this work.

General Training Configuration

All recurrent neural network (RNN) architectures were implemented using the PyTorch framework, a standard platform for deep learning research. Unless specified otherwise, models were trained using the stochastic gradient descent (SGD) optimizer with a momentum value of 0.9 and a small L2 regularization term (weight decay) of 10^(-7). SGD with momentum was chosen as it is a robust, well-understood optimization algorithm that is less prone to some of the instabilities that can arise with more adaptive methods when dealing with the complex, non-convex loss landscapes of RNNs.

A critical challenge when training RNNs on tasks with long delay intervals (i.e., T_delay >> T_resp) is the inherent imbalance in the loss function. Consider a task with a 400ms delay and a 50ms response period. The network is expected to output 0 for over 88% of the trial. A naive mean-squared error (MSE) loss, which averages the error across all time points equally, would be overwhelmingly dominated by this long "do nothing" period. This creates a strong local minimum for the optimizer: a trivial solution where the network simply learns to output 0 at all times. This achieves a low loss without actually learning the temporal aspect of the task.

To mitigate this fundamental issue, we trained all RNNs using a weighted MSE loss. This approach rebalances the contribution of different task epochs to the total loss, placing a much higher emphasis on getting the response correct during the brief but crucial reaction period. The weight function was defined as follows:

$$w_{MSE} = \begin{cases} 1 - \left(\frac{10}{T_{total}}\right), & if\ t \in T_{resp} \\ \frac{10}{T_{total}}, & otherwise \end{cases} \tag{3.15}$$

This weighting scheme ensures that the total error contributed by the short reaction period is comparable to the total error from the long delay period. This prevents the optimizer from ignoring the response period and forces it to grapple with the actual temporal challenge of the task, thereby preventing convergence to trivial solutions.

Large-Scale Experimental Design

In this study, we conducted a series of large-scale short-term memory experiments, totaling over 80,000 independent training runs, to systematically investigate the

underlying memory mechanisms across a wide range of conditions. We examined rank-2 RNNs on the delayed activation task to gain clear, visualizable insights, and then generalized these findings by training full-rank RNNs on the more complex delayed cue-discrimination task. We also explored the role of the neural time constant (τ) and the memory component itself by running several targeted control experiments. The configurations and total number of runs for each major experiment are summarized below.

Table 3.1. Number of Trainings on Large-Scale Experiments

RNN Type	**Task**	**Learning Rate (α)**	**Delay / Response Time**	**# Seeds**	**# Experiments**
Rank-2	Delayed Activation	5e-3	T_delay = 30 ms	100	1,600
Full	Cue-Discrimination (No Delay)	logspace(1e-3, 1) (20 vals)	T_resp = linspace(20, 400) ms (20 vals)	20	8,000
Full	Cue-Discrimination (Longer τ)	logspace(1e-3, 1e-1) (10 vals)	T_delay = linspace(0, 800) ms (21 vals)	20	4,200
Full	Cue-Discrimination (No T_post)	logspace(1e-3, 1e-1) (15 vals)	T_delay = linspace(0, 400) ms (21 vals)	100	31,500
Full	Cue-Discrimination (No T_post)	logspace(1e-1.6, 1e-0.5) (15 vals)	T_delay = linspace(0, 400) ms (21 vals)	100	31,500
Full	Cue-Discrimination (With T_post)	logspace(1e-3, 1e-1) (15 vals)	T_delay = linspace(0, 400) ms (21 vals)	20	6,300

Specific Parameters for Reproduction

To facilitate the reproduction of our key findings, the specific parameters for each set of experiments are detailed below. The full code and dataset are publicly available at github.com/fatihdinc/dynamical-phases-stm and via Zenodo.

Low-Rank RNN Experiments (Investigating Emergent Mechanisms):

To gain a clear, interpretable window into the emergent mechanisms, we began with low-rank RNNs. We chose a rank of K=2 because this is the minimum dimensionality required to support limit cycle oscillations, allowing us to observe the potential emergence of both slow-point and oscillatory solutions. These networks had N=100 neurons. The key parameters were:

- **Task:** Delayed Activation
- **Time constants:** τ = 10 ms, dt = 5 ms
- **Task timing:** T_delay = 150 ms, T_resp = 50 ms
- **Optimizer:** SGD with learning rate $\alpha = 10^{(-2)}$
- **Training:** 150,000 epochs
- **Initialization:** No noise (epsilon = 0) for clarity, firing rates from N(0, 0.1)

Task Modification Experiments (Effect of Post-Reaction Period):

To demonstrate the profound impact of task structure on the learned solution, we trained identical rank-2 RNNs (N=100, K=2) but added a brief post-reaction period (T_post = 30 ms) during which the network had to return to a zero output. We also varied the delay interval to observe how this interaction played out.

- **Task:** Delayed Activation with and without T_post
- **Task timing:** T_delay = {30, 40, 50, 60, 70} ms, T_resp = 10 ms
- **Optimizer:** SGD with learning rate $\alpha = 5 \times 10^{(-3)}$
- **Training:** 500,000 epochs to ensure convergence on these more complex temporal patterns.

Large-Scale Full-Rank RNN Experiments:

These experiments formed the core of our empirical validation, designed to test whether the principles discovered in low-rank models would generalize to more complex, unconstrained networks. Unless otherwise noted in the summary table, the primary training pipeline used the following fixed parameters:

- **Network:** N = 100 neurons (full-rank)
- **Task:** Delayed Cue-Discrimination
- **Time constants:** τ = 10 ms, dt = 5 ms
- **Task timing:** T_in = 30 ms, T_resp = 50 ms
- **Noise:** A small amount of noise (Epsilon drawn from $N(0, 10^{(-3)})$) was added to the dynamics to ensure robustness.
- **Initialization:** Firing rates from N(0, 0.1)
- **Training:** An adaptive schedule was used for the number of epochs: num_epochs = max(1000, 30 / α). This was crucial for a fair comparison, as networks trained with very low learning rates require more iterations to converge. The delay interval was varied from 0 to 420 ms in increments of 20 ms. Learning rates were sampled logarithmically to efficiently cover multiple orders of magnitude.

Control Experiment (Longer Time Constant):

To test whether our observed scaling laws were dependent on the specific intrinsic timescale of the neurons, we repeated the large-scale experiment but doubled the time

constant. A longer τ makes the neurons "slower" and more naturally able to integrate information over time.

- **Parameter change:** $\tau = 20$ ms
- **Delay range:** T_delay was correspondingly sampled from a longer range of 0 to 840 ms in increments of 40 ms.

Control Experiment (No Memory Component):

To definitively isolate the importance of the memory component (the delay period) in driving our results, we ran a crucial control experiment where we removed the delay entirely. The challenge was no longer to remember information over time, but simply to produce a sustained output.

- **Parameter change:** T_delay = 0 ms
- **Varied parameter:** The response interval, T_resp, was varied instead of the delay, from 20 to 400 ms.

Training: A fixed 3000 epochs was used.

Chapter 4:
RESULTS

4.1 Latent Mechanisms Underlying Neural Sequences

As a first step in our investigation, we set out to answer our first and most fundamental research question (**Q1: What mechanisms support memory maintenance through sequential neural activity?**). To do this, we employed a "minimalist" approach, using a simple short-term memory task that was stripped of all extraneous components like cue processing or decision-making, thereby isolating the core challenge of timing a delay. Specifically, we focused on the delayed activation task (Dinc et al., 2025b). In this task, the network is initialized to a particular state and must learn to do nothing—withhold its response (output = 0)—for a precise time window, T_delay. Immediately afterward, the network is required to produce a sustained response (output = 1) for a duration of T_resp. The task's simplicity is its primary virtue, as it forces the network to generate the required temporal pattern based purely on its own internal, self-generated dynamics.

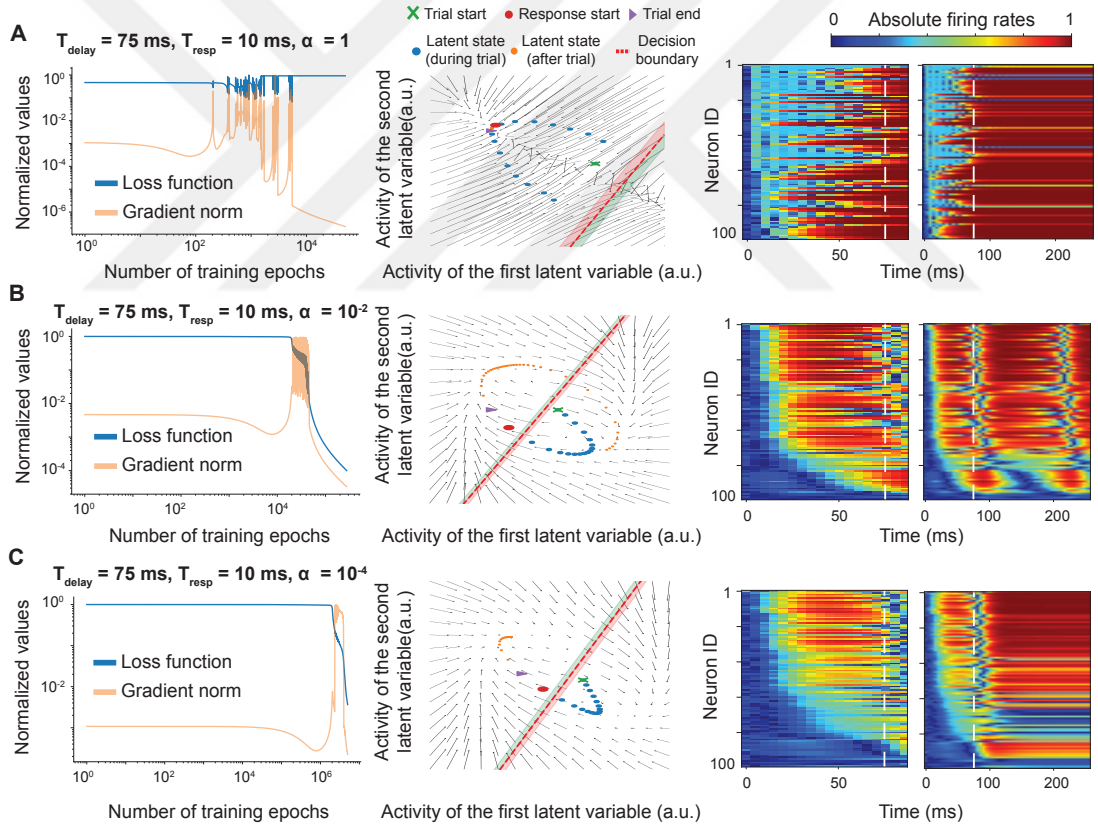

Figure 4.1. The learning rate of an RNN determines which attractor mechanism is learned for a delayed activation task.

Two distinct mechanisms can subserve neural sequences: Broadly, we hypothesized that two distinct classes of latent dynamical mechanisms could solve this task by generating the required neural sequences. The first mechanism relies on the generation of a set of **slow-points**, which are regions in the neural state space where the dynamics, or

the speed of the state's evolution, slow to a crawl ($dr(t)/dt \approx 0$). In this scenario, a neural sequence emerges as the system's state coasts slowly and predictably along a pre-defined path, or manifold, of these slow points. This transition is highly structured, meaning that from a given starting condition, the same latent trajectory is reliably followed (Sussillo & Barak, 2013). This stable, slow-moving trajectory can thus be used as a robust timer. The second, fundamentally different, mechanism relies on the generation of **limit cycles**. These are closed, circular trajectories in the state space that represent sustained, periodic oscillations. In this mechanism, the network learns to use the phase of its internal oscillation as a clock. When the learned period of the oscillation is much larger than the trial duration, the portion of the cycle traversed during the task appears as a non-repeating sequence. However, a key distinguishing feature is what happens after the task is over: a limit cycle, being a persistent oscillator, would continue to cycle and regenerate the sequence, whereas a slow-point manifold is a transient, one-way path, and the system's state would typically converge to a stable fixed point after the sequence is complete.

Since a minimum of two dimensions is required to generate oscillatory behavior in a dynamical system (Strogatz, 2018), we began by studying the learned latent representations in an RNN with a rank of K=2. This constraint allows us to directly visualize the complete two-dimensional latent dynamics. Our experiments revealed that the learned mechanism was a direct function of the optimization parameters. For the largest learning rate, the rank-2 RNN was incapable of solving the task; the optimization was unstable and failed to converge. As the learning rates were decreased into a workable range, we found that networks trained with a moderate learning rate consistently learned to solve the task by creating **limit cycles**. When the learning rate was decreased even further, to a very low value, the networks discovered a different solution: they learned to create **slow-point manifolds**. Notably, despite their different underlying dynamical structures, both mechanisms produced the required output and generated rich, sequential patterns of neural activity during the task window.

Despite leading to equivalent behaviors and neural activities within the trial window, the two mechanisms had qualitatively different and easily distinguishable behaviors after the trial concluded. The limit cycle solutions led to the recurrent, periodic generation of the same neural sequences, as the latent state continued to traverse its learned orbit. In contrast, the slow-point manifold solutions were transient; after generating the timed response, the network's state would typically exit the manifold and converge to a simple persistent activity state (a fixed point). This distinction is crucial, as it provides a clear, experimentally testable prediction. But first, it raises another question: is the choice of mechanism fixed from the start of training, or can it change?

Learned latent mechanisms can evolve during learning: To investigate this, we studied the evolution of the latent dynamical system of a rank-2 RNN over the course of a single training run. The learning curve (the loss function over training epochs) showed characteristic periods of rapid improvement followed by plateaus, which corresponded directly to qualitative shifts in the underlying mechanism. Specifically, early in training, after the first major drop in the loss function, the network had formed a rudimentary SP manifold. This "line" of slow points in the latent space was sufficient to solve the task approximately. However, as training continued, this linear manifold began to acquire a curved structure, suggesting an onset to a rotating, oscillatory solution. Finally, after a period of instability and oscillations in the loss function, the system settled into a new,

more stable solution: a well-formed limit cycle had emerged in the latent dynamical system. This confirmed that RNNs do not necessarily commit to a single strategy. Instead, the optimization process can dynamically explore the space of possible mechanisms, in this case transitioning from a less stable SP manifold solution to a more robust limit cycle solution.

This observation suggests that, from the perspective of the learning algorithm, there is an equivalence class of mechanisms that can solve the task. However, as we will show, these mechanisms are not equally easy to learn or equally stable, which creates a rich interplay between the task, the optimization, and the final learned solution.

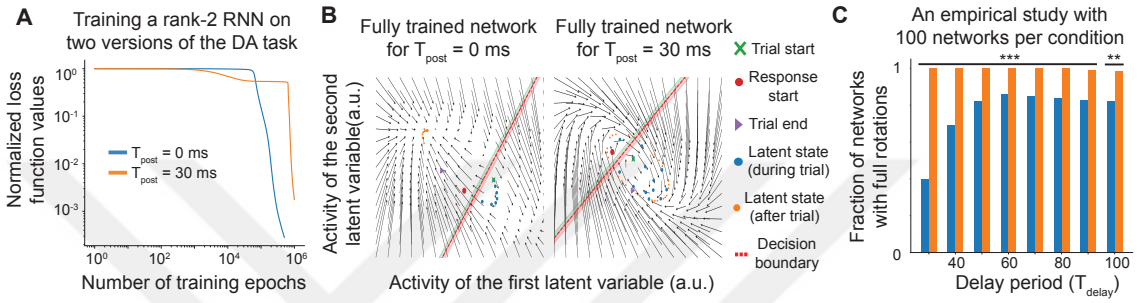

Figure 4.2. Minor modifications to task parameters strongly favor the learning of limit cycle solutions.

4.2 Influence of Task Design on the Learned Mechanism

Behavioral experiments in systems neuroscience often contain arbitrary design choices whose implications for the underlying neural computations are rarely scrutinized. Consider the delayed cue discrimination task. In the machine learning context, it consists of four distinct periods: cue, delay, reaction, and an optional post-reaction period. If a post-reaction period is included, the network must learn to turn its response off and return to a baseline state. In many animal experiments, however, the trial ends and the reward is delivered immediately after the response, so there is no explicit "off" period. This seemingly minor distinction, as we show below, can have a profound impact on the nature of the learned solution.

Minor modifications to the task requirements can change the learned mechanisms: We now set out to answer our second question (**Q2: What determines which mechanism is favored under different task or training conditions?**). To do so, we performed a simple but revealing manipulation: we took the delayed activation task and added a short post-reaction waiting period, requiring the network to switch its response off at the end of the reaction period and output zero for a brief duration. In networks trained without this modification (T_post = 0ms), the latent dynamical systems learned a mixture of solutions: slow-point manifolds for shorter delays and limit-cycles for longer delays. However, the result of adding even a brief post-reaction period (T_post = 30ms) was dramatic and unequivocal: the networks now predominantly and almost exclusively learned limit cycle solutions, despite achieving similar task performance. We quantified and confirmed this finding with a broader empirical study of 1600 RNNs, which showed a statistically significant shift in the distribution of learned mechanisms. The reason for this is dynamically intuitive: a limit cycle is a closed loop, an orbit that naturally brings

the system's state back towards its starting point. This makes it an ideal substrate for a task that requires a "reset" or a return to baseline. An SP manifold, in contrast, is an open-ended, transient path; it has no intrinsic mechanism for returning to the start. Thus, by adding a simple post-reaction requirement, the task design creates a strong inductive bias that favors the discovery of oscillatory solutions.

4.3 A Theoretical Scaling Analysis of Mechanistic Phases

Having established the existence of these two distinct mechanisms and their dependence on task structure, we next sought to answer our third question (**Q3: How do the learned mechanisms vary with delay duration?**). To do this, we turned to a set of low-dimensional analytical models to study their learning dynamics in a controlled, mathematical setting.

Why study low-dimensional toy models? Recent work has shown that even when trained without constraints, the dynamics of full-rank RNNs often converge to effectively low-dimensional solutions. In this view, training the millions of parameters in an RNN is equivalent to shaping a low-dimensional latent flow map of the form:

$$\tau \frac{d\kappa}{dt} = G(\kappa(t), u(t); W, W_{in}, b) \tag{4.1}$$

where G governs the evolution of a few latent variables kappa(t). Our toy models are designed to be the simplest possible systems that can implement the core logic of G for either a slow-point or a limit cycle. By studying how these simple systems learn, we can gain deep intuition about the fundamental constraints on the learning process, which may be universal across a wide range of model complexities.

Extracting scaling laws from toy models: Our detailed analysis, presented in the Appendix, reveals that the maximum stable learning rate follows a power law for both mechanisms, but with starkly different exponents.

- For the slow-point (SP) mechanism, which relies on fine-tuning the system to a bifurcation point, the maximum learning rate scales as: $\alpha_{SP} \sim O(T_{delay}^{-\beta_{SP}})$, where the exponent β_{SP} is between 4 and 5.
- For the limit cycle (LC) mechanism, which relies on tuning the frequency of an oscillator, the maximum learning rate scales as: $\alpha_{LC} \sim O(T_{delay}^{-\beta_{LC}})$ **,** where the exponent β_{LC} is between 2 and 3.

This is a central result of our work. The fact that β_{LC} is significantly smaller than β_{SP}means that the limit cycle solution is fundamentally more scalable and easier to learn. The learning difficulty for the SP mechanism grows extremely rapidly with the delay (as T^5), while for the LC mechanism, it grows much more moderately (as T^3). This provides a clear theoretical explanation for our earlier observations: as the delay increases, it becomes exponentially harder to find a stable SP solution, pushing the optimizer to find the more robust LC solution instead.

Further insights from toy models: The existence of these two distinct scaling laws highlights the existence of a "phase space" of learnable mechanisms. This phase diagram, with axes of delay time and learning rate, is partitioned into regions where only one

mechanism is learnable, or where both are, or where learning is impossible. This provides a powerful predictive framework for understanding the outcomes of our large-scale experiments.

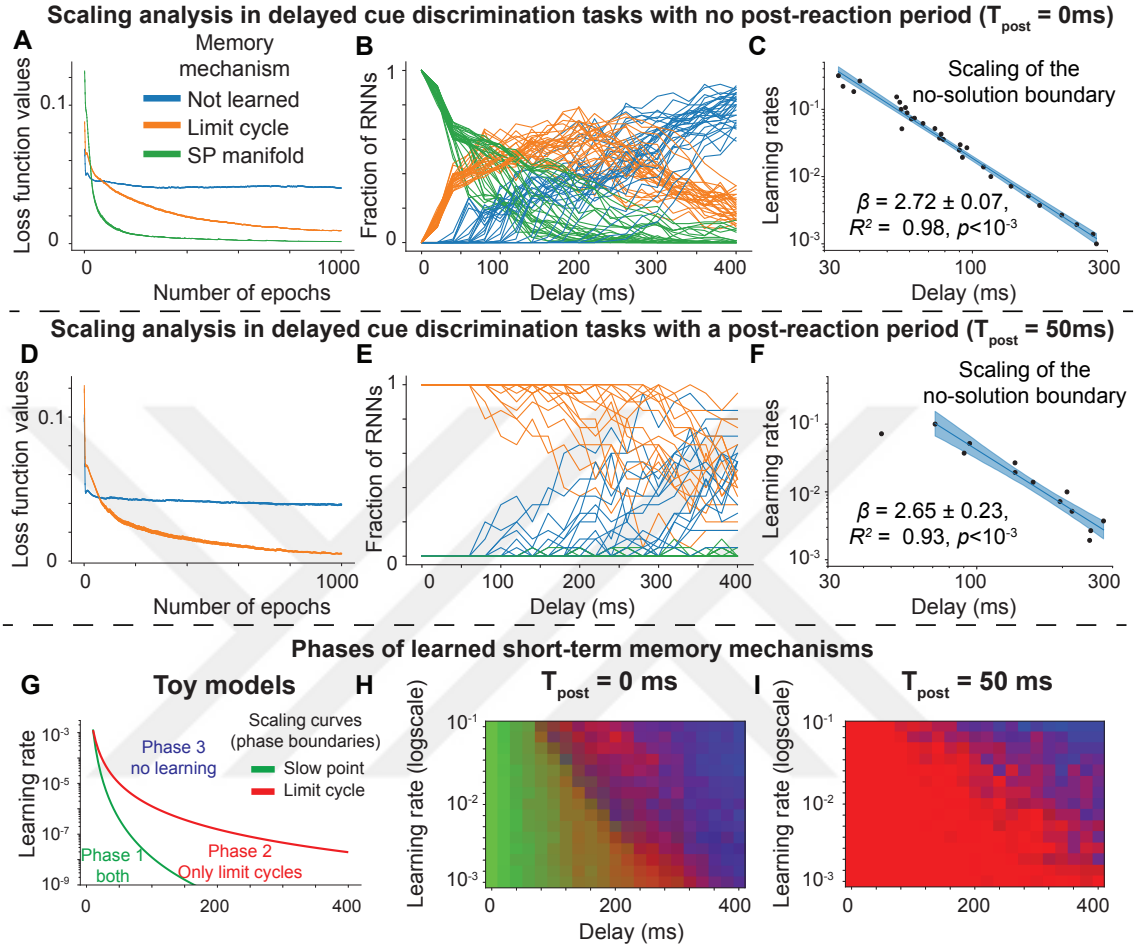

Figure 4.3.Empirical validation of theoretical predictions reveals emergent dynamical phases in full-rank RNNs.

4.4 Empirical Scaling Analyses of Latent Mechanisms

As a final and critical step, we tested the theoretical predictions derived from our toy models by training over 80,000 full-rank RNNs on the delayed cue-discrimination task. This allowed us to see if the principles uncovered in our simplified models would hold true in a more complex, high-dimensional, and biologically-plausible setting.

Large-scale RNN experiments reproduced the toy model predictions: The results were a strong confirmation of our theory. Using our Mechanism Discrimination Index (MDI), we automatically classified the outcome of each of the thousands of training runs. We found that, in line with our theoretical predictions, increasing the length of the delay period for a fixed learning rate caused a clear shift from slow-point solutions to limit cycle solutions. We then extracted the empirical scaling laws for the boundaries of these phases. The quantitative agreement was remarkable:

- The empirical scaling exponent for the boundary of **limit cycle** solutions was β_{LC} = **2.72 ± 0.07**, in close agreement with our theoretical prediction of 3.
- The empirical scaling exponent for the boundary of **slow-point** solutions was β_{SP} = **4.05 ± 0.10**, also matching our theoretical range of 4-5.

Furthermore, when we trained RNNs on the task variant that included a post-reaction window, we confirmed that slow-point solutions were almost entirely eliminated, and only oscillatory dynamics emerged, as predicted. We summarized these findings by plotting the empirical phase diagrams, which showed a striking resemblance to the theoretical diagram derived from the toy models, capturing the precise conditions under which each mechanism emerged.

Control experiments: Finally, we performed two crucial control experiments to solidify our conclusions. First, we repeated the analysis using RNNs with a larger, slower intrinsic time constant (larger τ). This did not alter the observed scaling laws, demonstrating that these laws are a fundamental property of the *learning dynamics*, not just the specific timescale of the neurons. Second, and most importantly, we tested a version of the task that no longer required memory maintenance (T_{delay} = 0). In this case, the strong dependence on task duration nearly vanished, and the scaling was substantially weakened (**beta = 0.38 ± 0.02**). This finding provides definitive evidence that the computational mechanisms and their associated scaling laws are a direct consequence of the challenge of storing information over a temporal delay. It is the memory requirement itself that shapes the landscape of possible solutions.

4.5 Experiments and Figure Details

Figure 2.1: (A-C) Visualizations of different neural mechanisms for short-term memory maintenance. **(A)** The classic persistent activity model posits that distinct memories (e.g., red vs. blue) are encoded by the sustained firing of separate neural populations [6, 7] **(B)** An alternative, sequential activation model suggests that memory can be stored in the ordered firing of a single neural population, with different readout neurons decoding the memory content [8] **(C)** This work [37] demonstrates that limit cycles (periodic orbits) can generate activity that locally approximates the sequential mechanism in (B). However, unlike transient sequences, limit cycles will continue to oscillate, potentially causing spurious periodic activity in a post-reaction period (magenta). **(D-E)** The delayed cue-discrimination task and its implementation for RNNs. **(D)** The task consists of four stages: (i) a cue presentation, (ii) a delay period, (iii) a reaction period, and (iv) an optional post-reaction period. **(E)** For RNNs, the task is adapted by mapping two input channels to two output channels. A brief input pulse on one channel must be held in memory through the delay and reproduced on the corresponding output channel. When the initial cue period is removed, the task is referred to as the delayed activation task.

Figure 4.1: We trained a rank-2 recurrent neural network (RNN) to perform a delayed activation task, where the network must withhold its output for a duration of T_delay before producing a response for T_resp. The network's activity is unconstrained outside of this task window. Each row corresponds to a different learning rate. (A) With a high learning rate (α = 1), training is unstable, and the network fails to learn the task. The left panel shows the volatile loss and gradient norm during training. The middle panel shows the latent dynamics, which do not converge to a useful solution. The right panel shows

the resulting disorganized neural firing rates. (B) Using a moderate learning rate ($\alpha = 10^{-2}$), the network successfully solves the task by forming a limit cycle attractor. The latent dynamics (middle) converge to a stable periodic orbit that times the delay. The corresponding neural activity (right) shows a robust, sequential pattern that becomes periodic after the trial. (C) With a low learning rate ($\alpha = 10^{-4}$), the network again solves the task but discovers a different solution: a slow-point manifold. The latent dynamics (middle) follow a transient, non-repeating trajectory that slows down to time the delay before settling at a fixed point. For all panels, the middle column shows the flow map of the two latent variables, with shaded regions indicating the decision boundary for the output (between 0.25 and 0.75). The right column shows the absolute firing rates of all 100 neurons, with the dotted line marking the start of the response period. General parameters: N=100 neurons, τ=10ms, Δt=5ms, T_delay=75ms, T_resp=10ms. (Paszke et al., 2017).

Figure 4.2: This figure demonstrates how adding a post-reaction period to the delayed activation task biases the learned computational mechanism. (A) Comparison of the learning curves for two task versions: one without a post-reaction period (T_post = 0ms, blue) and one with an added 30ms post-reaction period (orange). Both networks learn the task successfully. (B) The learned latent dynamics for the fully trained networks from panel (A). The network trained on the standard task converges to a transient slow-point manifold. In contrast, the network trained with the post-reaction requirement discovers a periodic limit cycle solution. (C) A large-scale study generalizing this finding. This panel shows the fraction of networks (out of 100 per condition) that developed limit cycle solutions when trained on the standard task (blue) versus the task with a post-reaction period (orange), as a function of the delay duration (T_delay). The addition of a post-reaction period consistently and significantly increases the probability of learning a limit cycle. *Parameters for (A-B): As in Figure 2, with T_delay = 30ms, T_resp = 10ms, and $\alpha = 10^{-3}$. Parameters for (C): $\alpha = 0.005$. Statistical comparisons were made using Fisher's exact test with Bonferroni correction (***$p < 0.001$, *$p < 0.01$).

Figure 4.3: This figure presents the results of large-scale experiments training full-rank RNNs on the delayed cue-discrimination task, confirming the scaling laws and phase transitions predicted by the analytical models. (A-C) Task without a post-reaction period: (A) Representative loss curves for the four learning outcomes: successful convergence to a limit cycle (green) or slow-point manifold (orange), and failure due to instability (red) or non-convergence (blue). (B) The distribution of learned mechanisms as a function of delay duration. Each line represents a different learning rate, showing that for any given rate, learning becomes more difficult and favors limit cycles as the delay increases. (C) The empirical scaling law for the no-solution boundary. Each point represents the maximum delay learnable at a given learning rate. The log-log plot reveals a clear power-law relationship, from which the critical exponent is extracted. (D-F) Task with a post-reaction period: These panels show the same analyses as (A-C) for the task variant that includes a post-reaction window. The results show a dramatic reduction in the prevalence of slow-point solutions. (G-I) Theoretical and Empirical Phase Diagrams: (G) The theoretical phase diagram predicted by our analytical toy models, showing distinct regions where different mechanisms are expected to be learnable. (H) The empirical phase diagram for the standard task, constructed from the data in (B). The color intensity shows the proportion of networks converging to each state. The diagram shows three distinct phases—slow-point, limit cycle, and no learning—that closely match the theoretical

prediction. (I) The empirical phase diagram for the task with a post-reaction period. As predicted, the slow-point phase is almost entirely eliminated, biasing the network to learn only limit cycle dynamics. These phase diagrams were generated from over 65,000 trained RNNs, varying learning rates, delay intervals, and random seeds. For full parameter details, see the Methods and Reproduction sections.

Chapter 5:
CONCLUSION

In this work, we set out to answer three fundamental questions about the population-level mechanisms that can support short-term memory in neural networks. By combining theoretical analysis of simplified models with large-scale empirical simulations of more complex recurrent neural networks, we have developed a cohesive framework for understanding the computational principles, constraints, and trade-offs that govern the learning of memory-dependent tasks. Our findings provide a bridge between the abstract language of dynamical systems theory and the practical challenges of training neural networks, offering insights that are relevant to both artificial intelligence and neuroscience. Here, we synthesize our findings by revisiting our initial research questions and discussing their broader implications.

5.1 Mechanisms for Sequential Memory

What mechanisms support memory maintenance through sequential neural activity?

In the memory tasks studied here, we found that neural activation sequences consistently emerged as the core component of the learned solutions. Our work identified and characterized two distinct and robust dynamical mechanisms capable of generating these sequences: **slow-point manifolds** and **limit cycles**.

The use of slow-points to generate transient, non-repeating sequences is well aligned with prior intuitions in the field [38]. This mechanism is an elegant solution that leverages the geometry of the state space. By learning to configure its weights to be infinitesimally close to a saddle-node bifurcation, the network creates a "ghost" of a fixed point. This ghost manifests as a "channel" of slowed dynamics through which the network state can reliably coast for a precise duration. It is an elegant and efficient solution for one-shot timing tasks, creating a transient, self-terminating sequence without requiring any oscillatory machinery. Its transient nature is both a strength and a weakness: it is perfectly suited for a single, timed event, but it is not intrinsically designed for repetition or for returning the system to its initial state.

However, the spontaneous emergence of limit cycles in this context represents a novel and significant prediction of our framework. Although limit cycles have been previously observed in RNNs trained on memory tasks [39], they typically appear when periodicity is explicitly or implicitly embedded in the task design (e.g., rhythmic inputs or required outputs). In contrast, the delayed activation and delayed cue-discrimination tasks used here had no explicit periodic components. The network's discovery that a long-period oscillation is a robust and efficient way to create a reproducible, timed trajectory is a non-trivial finding. It suggests an alternative, previously underappreciated mechanism for generating sustained neural sequences in the absence of any external rhythmic input or an oscillatory output requirement. This solution co-opts a periodic dynamical structure for an aperiodic task, using the phase of the oscillation as a reliable internal clock. This strategy has the inherent advantage of being an attractor: the network state is robustly pulled towards the circular trajectory, making the resulting timing resistant to noise and perturbations.

5.2 Determinants of the Learned Mechanism

What determines which mechanism is favored under different memory tasks or training conditions?

Our analytical results revealed that the selection between slow-point manifolds and limit cycles was not a random outcome of the optimization process, but was a predictable result determined by the interplay between the structure of the task and the learning rate used for training. We later confirmed these theoretical predictions in our large-scale simulations using full-rank RNNs, demonstrating that these principles generalize from simple models to more complex networks.

This finding is particularly striking in light of the recent and growing interest in dynamical systems approaches in neuroscience, where considerable effort has gone into identifying and manipulating the attractors and manifolds that are thought to underlie neural activity and behavior [40, 41]. Our results show a sharp divergence in the learned dynamical solutions—from transient slow-points to periodic limit cycles—emerging from minor, seemingly innocuous changes in task design. This underscores a critical and often overlooked point: the structure of the task can strongly shape, and in some cases dictate, the neural dynamics that emerge. The loss landscape is not fixed; it is molded by the task's constraints, and the optimizer will find the path of least resistance through that landscape.

This raises important and challenging questions about how we interpret dynamical structures observed in neural recordings. Can we conclude that one type of solution (e.g., a line attractor) is a more fundamental or "correct" representation of a cognitive process than another, simply because it appears in a specific task variant? Our findings highlight the need for profound caution. They suggest that conclusions about the nature of neural computations must be grounded not only in the observed dynamics but also in a careful theoretical analysis of how the task parameters may constrain or bias the space of possible solutions. What may appear to be a core computational motif—a "neural signature" of short-term memory, for instance—could, in fact, reflect a contingent outcome of the task design itself. For example, an experiment that implicitly requires an animal to "reset" between trials might strongly favor the observation of oscillatory dynamics, leading to the potentially erroneous conclusion that the brain *always* uses oscillations for that cognitive function.

5.3 Scaling Laws and the Limits of Memory

How do the learned mechanisms vary with the delay duration?

A key insight into this question comes from our derivation and empirical validation of scaling laws that govern the learning process. We found that the critical learning rate—the maximum rate allowing for successful training—decreases as a power-law function of the required delay. As the delay increases, smaller and smaller parameter updates are required to fine-tune the network's temporal dynamics, ultimately making training impractically slow and unstable for very long durations. This constraint provides a principled, dynamical systems explanation for the historical difficulty of capturing long-term dependencies in RNNs, a problem first highlighted in the seminal work of Bengio et al. [2]. The reason for the different scaling exponents is also intuitive: creating a slow

point requires precisely tuning the system to the edge of a bifurcation, a task that demands exponentially increasing precision. In contrast, creating a limit cycle only requires tuning a frequency, a less demanding task.

The introduction of gated architectures such as LSTMs and GRUs partially addressed this issue by enabling networks to learn adaptive, input-dependent timescales [3]. These architectures could, in principle, bypass the rigid power-law constraint by using their gating mechanisms to effectively "freeze" parts of their internal state, protecting information from the decay and interference inherent in simpler RNN dynamics. However, this flexibility introduces its own trade-offs. Extending timescales indefinitely slows down a network's ability to react to new information, limiting its utility for rapid decision-making. Furthermore, expanding the parameter space to accommodate learnable timescales can complicate the optimization landscape and destabilize learning, leading to its own set of challenges [36]. Unlike biological systems, which operate under relatively fixed biophysical constraints such as membrane time constants, artificial networks must learn these properties through training—a process that is not guaranteed to succeed even if the underlying network is theoretically capable of solving the task.

Overall, by framing the challenge of short-term memory through the lens of dynamical systems theory, our work highlights a core, and likely universal, trade-off between memory duration, learning speed, and network stability. Understanding this trade-off offers a principled explanation for the persistent difficulty of learning long-term dependencies and opens the door to new strategies that seek to balance biological plausibility, computational efficiency, and learning performance.

5.4 Concluding Remarks and Future Directions

Our findings carry important implications for both experimental neuroscience and machine learning. We have shown that both slow-points and limit cycles can serve as effective and robust mechanisms for generating the sequential neural activity that underlies short-term memory, and we have characterized the conditions under which each is likely to emerge.

For neuroscience, a key challenge remains: in the context of a dynamic brain, where synaptic plasticity continually reshapes functional connectivity [42], how can we experimentally determine whether an observed neural sequence is supported by a transient slow-point manifold or a periodic limit cycle? More broadly, how can we distinguish between dynamical solutions that produce equivalent activity patterns *during* a trial, but diverge in their behavior once the trial ends? To our knowledge, this question—concerning the equivalence of distinct latent mechanisms subserving task execution—has not been explicitly addressed in experimental designs within systems neuroscience. Tackling it would require a shift in how experiments are conceptualized and conducted. As a starting point, our work illustrates how even minor manipulations of task structure, such as observing neural activity in a prolonged post-trial period, can lead to distinct and identifiable underlying mechanisms. An experiment that randomly extends the post-response period could reveal whether the population trajectory reliably returns to a fixed point (suggesting a slow-point manifold) or continues to evolve along a closed loop (suggesting a limit cycle).

Beyond neuroscience, the phenomenon of staged learning dynamics and algorithmic phase transitions has been widely documented in modern artificial neural networks, including Transformers trained on tasks like modular arithmetic [43, 44], language modeling [45, 46], and in-context learning [47]. Within this growing body of literature, our study of RNNs provides a complementary perspective, grounded in the formal language of dynamical systems theory, that connects computational behavior to interpretable latent mechanisms. Recent efforts have aimed to formalize learning progress in feedforward networks using quantities analogous to order parameters in physics [48]. Our results go a step further by identifying literal phase transitions between qualitatively distinct computational strategies (attractor states) in dynamical systems. Future work may explore how these mechanisms, and the principles governing their emergence, can be implemented or approximated by more complex feedforward architectures, potentially unifying our understanding of learning dynamics across different classes of neural networks. Understanding these dynamical phases and their boundaries could also inform the development of more efficient training curricula or optimization algorithms that can navigate these complex landscapes more effectively.

BIBLIOGRAPHY

[1] C. Constantinidis and T. Klingberg, "The neuroscience of working memory capacity and training.," *Nature Reviews Neuroscience,* pp. 438-449, 2016.

[2] Y. Bengio, S. Patrice and F. Paolo, "Learning long-term dependencies with gradient descent is difficult.," *IEEE transactions on neural networks,* pp. 157-166, 1994.

[3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation, MIT Press,* pp. 1735-1780, 1997.

[4] J. Chung, C. Gulcehre, K. Cho and Y. Bengio, *Empirical evaluation of gated recurrent neural networks on sequence modeling,* arXiv preprint arXiv:1412.3555, 2014.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, ... and I. Polosukhin, "Attention is all you need," 2017.

[6] J. M. Fuster and G. E. Alexander, "Neuron activity related to short-term memory," *Science,* vol. 173(3997), pp. 652-654, 1971.

[7] R. C. Atkinson and R. M. Shiffrin, "Human memory: A proposed system and its control processes. In Psychology of learning and motivation," *In Psychology of learning and motivation,* vol. 2, pp. 89-195, 1968.

[8] K. Rajan, C. D. Harvey and D. W. Tank, "Recurrent network models of sequence generation and memory," *Neuron,* vol. 90(1), pp. 128-142, 2016.

[9] P. Jonas, G. Major and B. Sakmann, "Quantal components of unitary EPSCs at the mossy fibre synapse on CA3 pyramidal cells of rat hippocampus," *The Journal of physiology,* vol. 472(1), pp. 615-663, 1993.

[10] D. A. McCormick, B. W. Connors, J. W. Lighthall and D. A. Prince, "Comparative electrophysiology of pyramidal and sparsely spiny stellate neurons of the neocortex," *Journal of neurophysiology,* vol. 54(4), pp. 782-806, 1985.

[11] R. Miles, "Synaptic excitation of inhibitory cells by single CA3 hippocampal pyramidal cells of the guinea-pig in vitro," *The Journal of physiology,* vol. 428(1), pp. 61-77, 1990.

[12] M. Isokawa, "Membrane time constant as a tool to assess cell degeneration," *Brain Research Protocols,* vol. 1(2), pp. 114-116, 1997.

[13] J. R. Geiger, J. Lübke, A. Roth, M. Frotscher and P. Jonas, "Submillisecond AMPA receptor-mediated signaling at a principal neuron–interneuron synapse," *Neuron,* vol. 18(6), pp. 1009-1023, 1997.

[14] M. W. Bondi, E. C. Edmonds and D. P. Salmon, "Alzheimer's disease: past, present, and future," *Journal of the International Neuropsychological Society,* Vols. 23(9-10), pp. 818-831, 2017.

[15] J. Lee and S. Park, "Working memory impairments in schizophrenia: a meta-analysis," *Journal of abnormal psychology,* vol. 114(4), p. 599, 2005.

[16] J. Scott, G. Matt, K. Wrocklage, C. Crnich, J. Jordan, S. Southwick, J. Krystal and B. Schweinsburg, "A quantitative meta-analysis of neurocognitive functioning in posttraumatic stress disorder," *Psychological bulletin,* vol. 141(1), p. 105, 2015.

[17] J. J. Todd and R. Marois, "Capacity limit of visual short-term memory in human posterior parietal cortex," *Nature,* vol. 428(6984), pp. 751-754, 2004.

[18] C. E. Curtis and M. D'Esposito, "Persistent activity in the prefrontal cortex during working memory," *Trends in cognitive sciences,* vol. 7(9), pp. 415-423, 2003.

[19] S. A. Harrison and F. Tong, "Decoding reveals the contents of visual working memory in early visual areas," *Nature,* vol. 458(7238), pp. 632-635, 2009.

[20] J. W. Gnadt and R. A. Andersen, "Memory related motor planning activity in posterior parietal cortex of macaque," *Experimental brain research,* vol. 70(1), pp. 216-220, 1988.

[21] E. H. Baeg, Y. B. Kim, K. Huh, I. Mook-Jung, H. T. Kim and M. W. Jung, "Dynamics of population code for working memory in the prefrontal cortex," *Neuron,* vol. 40(1), pp. 177-188, 2003.

[22] S. E. Cavanagh, J. P. Towers, J. D. Wallis, L. T. Hunt and S. W. Kennerley, "Reconciling persistent and dynamic hypotheses of working memory coding in prefrontal cortex," *Nature communications,* vol. 9(1), p. 3498, 2018.

[23] E. M. Meyers, D. J. Freedman, G. Kreiman, E. K. Miller and T. Poggio, "Dynamic population coding of category information in inferior temporal and prefrontal cortex," *Journal of neurophysiology,* vol. 100(3), pp. 1407-1419, 2008.

[24] G. Mongillo, O. Barak and M. Tsodyks, "Synaptic theory of working memory," *Science,* vol. 319(5869), pp. 1543-1546, 2008.

[25] M. G. Stokes, "'Activity-silent'working memory in prefrontal cortex: a dynamic coding framework," *Trends in cognitive sciences,* vol. 19(7), pp. 394-405, 2015.

[26] E. H. Baeg, Y. B. Kim, K. Huh, I. Mook-Jung, H. T. Kim and M. W. Jung, "Dynamics of population code for working memory in the prefrontal cortex," *Neuron,* vol. 40(1), pp. 177-188, 2003.

[27] C. D. C. P. &. T. D. W. Harvey, "Choice-specific sequences in parietal cortex during a virtual-navigation decision task," *Nature,* vol. 484(7392), pp. 62-68, 2012.

[28] E. Spaak, K. Watanabe, S. Funahashi and M. G. Stokes, "Stable and dynamic coding for working memory in primate prefrontal cortex," *Journal of neuroscience,* vol. 37(27), pp. 6503-6516, 2017.

[29] V. Mante, D. Sussillo, K. V. Shenoy and W. T. Newsome, "Context-dependent computation by recurrent dynamics in prefrontal cortex," *Nature,* vol. 503(7474), pp. 78-84, 2013.

[30] D. Sussillo and O. Barak, "Opening the black box: low-dimensional dynamics in high-dimensional recurrent neural networks," *Neural computation,* vol. 25(3), pp. 626-649, 2013.

[31] A. Finkelstein, L. Fontolan, M. N. Economo, N. Li, S. Romani and K. Svoboda, "Attractor dynamics gate cortical information flow during decision-making," *Nature neuroscience,* vol. 24(6), pp. 843-850, 2021.

[32] A. Dubreuil, A. Valente, M. Beiran, F. Mastrogiuseppe and S. Ostojic, "The role of population structure in computations through neural dynamics," *Nature neuroscience,* vol. 25(6), pp. 783-794, 2022.

[33] A. Valente, J. W. Pillow and S. Ostojic, "Extracting computational mechanisms from neural data using low-rank RNNs," *Advances in Neural Information Processing Systems,* vol. 35, pp. 24072-24086, 2022.

[34] F. Dinc, M. Blanco-Pozo, D. Klindt, F. Acosta, Y. Jiang, S. Ebrahimi, A. Shai, H. Tanaka, P. Yuan, M. Schnitzer and N. Miolane, *Latent computing by biological neural networks: A dynamical systems framework,* arXiv preprint arXiv:2502.14337., 2025.

[35] F. Dinc, A. Shai, M. Schnitzer and H. Tanaka, "CORNN: Convex optimization of recurrent neural networks for rapid inference of neural dynamics," *Advances in Neural Information Processing Systems,* vol. 36, pp. 51273-51301, 2023.

[36] F. Dinc, E. Cirakman, Y. Jiang, M. Yuksekgonul, M. J. Schnitzer and H. Tanaka, *A ghost mechanism: An analytical model of abrupt learning,* arXiv preprint arXiv:2501.02378, 2025.

[37] B. Kurtkaya, F. Dinc, M. Yuksekgonul, M. Blanco-Pozo, E. Cirakman, M. Schnitzer, Y. Yemez, H. Tanaka, P. Yuan and N. Miolane, "Dynamical phases of short-term memory mechanisms in RNNs," *Forty-second International Conference on Machine Learning,* 2025.

[38] M. &. F. I. R. Khona, "Attractor and integrator networks in the brain," *Nature Reviews Neuroscience,* vol. 23(12), pp. 744-766, 2022.

[39] M. Pals, J. H. Macke and O. Barak, "Trained recurrent neural networks develop phase-locked limit cycles in a working memory task," *PLOS Computational Biology,* vol. 20(2), no. e1011852, 2024.

[40] M. Liu, A. Nair, N. Coria, S. W. Linderman and D. J. Anderson, "Encoding of female mating dynamics by a hypothalamic line attractor," *Nature,* vol. 634(8035), pp. 901-909, 2024.

[41] A. Vinograd, A. Nair, J. H. Kim, S. W. Linderman and D. J. Anderson, "Causal evidence of a line attractor encoding an affective state," *Nature,* vol. 634(8035), pp. 910-918, 2024.

[42] S. Ebrahimi, J. Lecoq, O. Rumyantsev, T. Tasci, Y. Zhang, C. Irimia, ... and M. J. Schnitzer, "Emergent reliability in sensory cortical coding and inter-area communication," *Nature,* vol. 605(7911), pp. 713-721, 2022.

[43] Z. Liu, E. J. Michaud and M. (. Tegmark, *mnigrok: Grokking beyond algorithmic data,* arXiv preprint arXiv:2210.01117, 2022.

[44] N. Nanda, L. Chan, T. Lieberum, J. Smith and J. Steinhardt, *Progress measures for grokking via mechanistic interpretability,* arXiv preprint arXiv:2301.05217, 2023.

[45] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, ... and W. Fedus, *Emergent abilities of large language models,* arXiv preprint arXiv:2206.07682, 2022.

[46] E. S. Lubana, K. Kawaguchi, R. P. Dick and H. Tanaka, *A percolation model of emergence: Analyzing transformers trained on a formal language,* arXiv preprint arXiv:2408.12578, 2024.

[47] A. Raventós, M. Paul, F. Chen and S. Ganguli, "Pretraining task diversity and the emergence of non-bayesian in-context learning for regression," *Advances in neural information processing systems,* vol. 36, pp. 14228-14246, 2023.

[48] C. F. Park, E. S. Lubana, I. Pres and H. Tanaka, *Competition dynamics shape algorithmic phases of in-context learning,* arXiv preprint arXiv:2412.01003, 2024.

Chapter 6:
ADDITIONAL FIGURES AND CAPTIONS

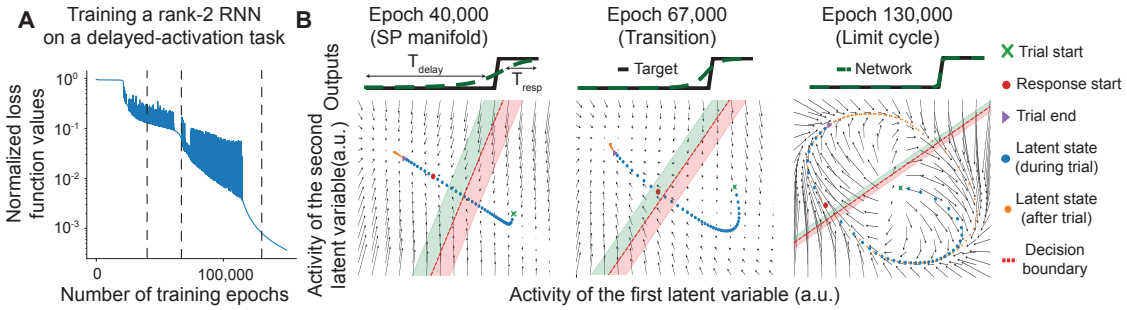

Figure 6.1. A recurrent neural network can transition between distinct memory mechanisms during a single training run.

Figure 7.1: This figure illustrates the evolution of the learned latent dynamics of a rank-2 RNN as it trains on the delayed activation task. **(A)** The loss function over 150,000 training epochs. The curve shows an initial rapid decrease, followed by a period of instability and a final convergence to a stable, low-loss solution. Dotted lines indicate the specific epochs analyzed in panel (B). **(B)** Snapshots of the two-dimensional latent dynamical system at three key stages of training, revealing a shift in the underlying computational strategy.

- **Left (Epoch 40,000):** Early in training, the network discovers an initial solution by forming a **slow-point (SP) manifold**. This line of slowed dynamics allows the network to time the delay. After the trial, the latent state converges to a final fixed point.
- **Middle (Epoch 67,000):** During a transitional phase, the SP manifold begins to curve and dissolve, suggesting the emergence of a rotational dynamic.
- **Right (Epoch 130,000):** In the final converged state, the network has abandoned the SP manifold in favor of a stable **limit cycle**. This periodic orbit provides a more robust timing mechanism. After the trial, the latent state continues to evolve in a periodic manner.

Parameters: N=100 neurons, τ=10ms, Δt=5ms, T_delay=150ms, T_resp=50ms, $\alpha=10^{-2}$. Shaded areas represent the output decision boundary (between 0.25 and 0.75). (Paszke et al., 2017).

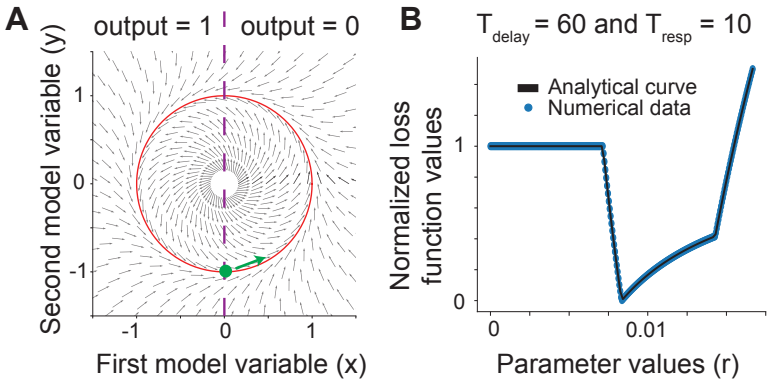

Figure 6.2. Theoretical analysis of toy models reveals distinct scaling laws and a resulting phase space of mechanisms.

Figure 7.2: **(A) The Limit Cycle Mechanism:** The flow map for the two-dimensional limit cycle model (with r=0.1). Trajectories are drawn towards a stable, attractive limit cycle (red circle). The system's state (starting at the green circle) rotates around this cycle, and its position relative to the decision boundary (purple line) determines the output. **(B) The Loss Landscape:** An example loss curve for the limit cycle model as a function of its frequency parameter r. The analytical derivation (blue line) perfectly matches numerical simulations (gray dots). The sharp "kink" at the global minimum is the key feature that constrains the maximum stable learning rate.

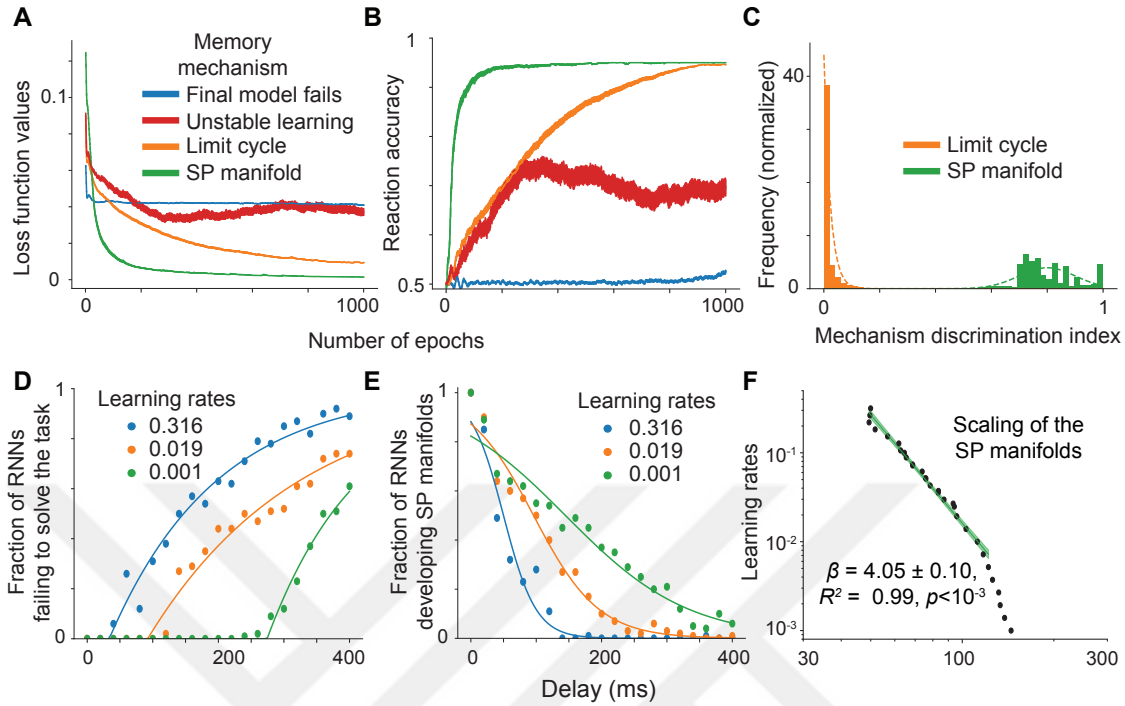

Figure 6.3. Quantitative analysis of learning outcomes and scaling laws in the large-scale RNN experiment.

Figure 7.3: This figure illustrates the quantitative steps used to classify learning outcomes and extract scaling laws from our large-scale experiments.
(A-B) First, we distinguished successful from unsuccessful training runs. Networks that learned a solution, whether a **limit cycle** or a **slow-point manifold**, showed stable convergence to a low loss (A) and achieved high reaction accuracy (B). In contrast, failed runs were characterized by high loss, instability, or poor accuracy.

(C) For the successful runs, we used the **Mechanism Discrimination Index (MDI)** to automatically classify the underlying strategy. This metric reliably differentiates the two solution types by quantifying the frequency content of their post-trial dynamics, even when their in-trial behavior is similar.

(D-E) Our analysis revealed a dual role for the learning rate. As expected, lower learning rates enabled networks to solve tasks with longer delays (D). More interestingly, lower learning rates also increased the likelihood that the network would converge to a slow-point manifold solution instead of a limit cycle (E).

(F) Finally, we extracted the empirical scaling law for the slow-point mechanism. As predicted by our analytical models, the relationship between the maximum learnable delay and the learning rate follows a distinct power-law. The fitted exponent of $\beta = 4.05 \pm 0.10$ falls squarely within our theoretically predicted range of 4 to 5. The data also show that these solutions are preferentially suppressed at longer delays ($T_{delay} > 100$ms), which is consistent with the idea that the optimization process favors the more scalable limit cycle mechanism in that regime.

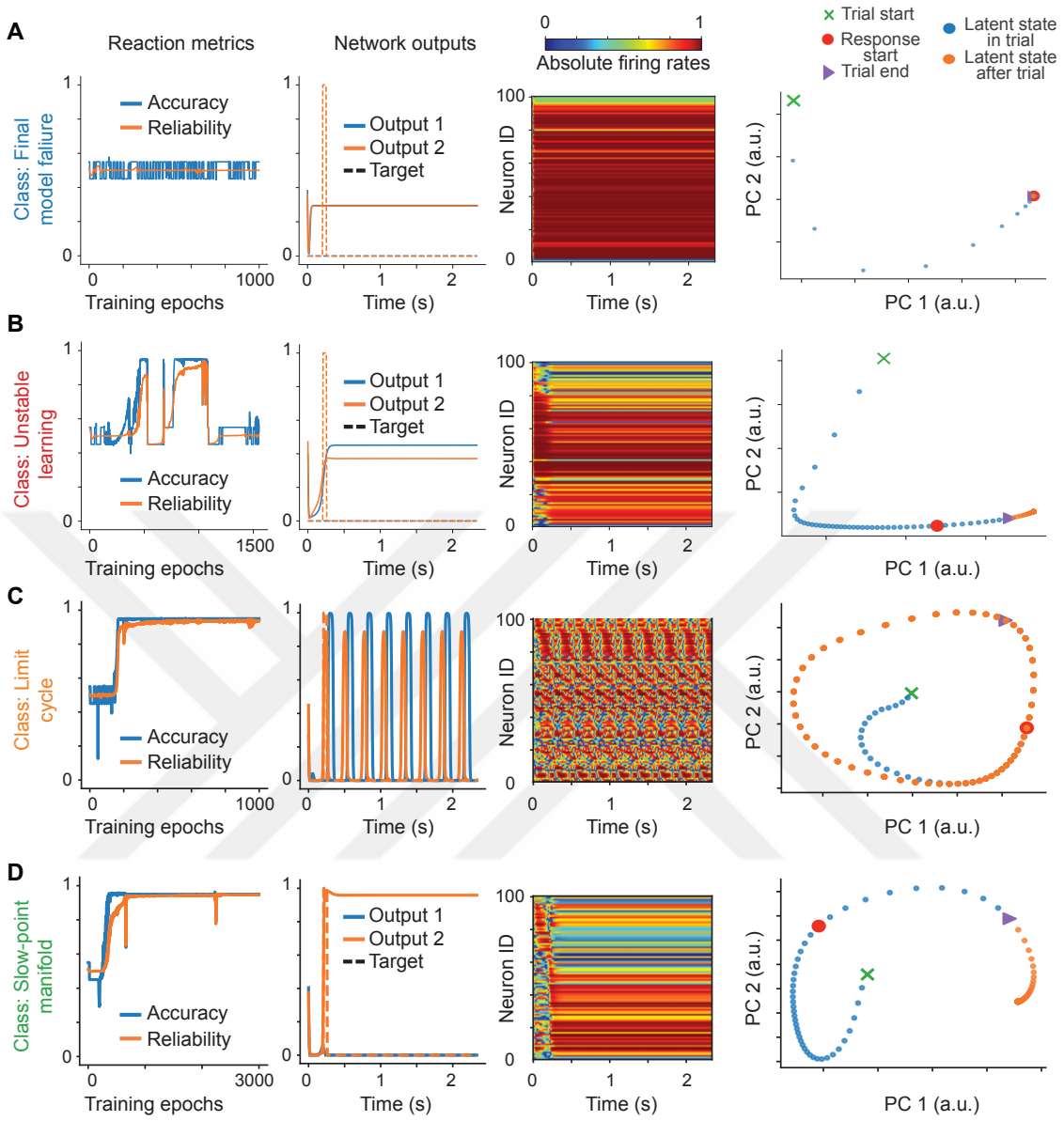

Figure 6.4. Representative examples of the four primary classes of learning outcomes.

Figure 7.4: This figure provides a visual guide to the four main classifications identified in our large-scale RNN experiment: (A) Final model failure, (B) Unstable learning, (C) Limit cycle, and (D) Slow-point manifold. All examples are from networks trained on a task with a 180ms delay, a condition chosen because it readily exhibits all four outcome types.

For each class, four panels are shown, from left to right:

1. **Reaction Metrics:** Reaction Accuracy and Reliability over the course of training.
2. **Network Outputs:** The final output of the trained network (blue/orange) compared to the ground truth signal (dashed).
3. **Firing Rates:** The activity of all 100 individual neurons, sorted by peak activation time.

4. **Latent Dynamics:** The trajectory of the first two principal components of the neural activity, revealing the underlying dynamical structure of the solution.

Specific training parameters for each example: (A) α=0.1, seed=93; (B) α≈0.0193, seed=41; (C) α=0.1, seed=82; (D) α=0.01, seed=85. Shared parameters: T_in=30ms, T_delay=180ms, T_resp=50ms, Δt=5ms, τ=10ms, σ_noise=0.001. See Methods for full details.

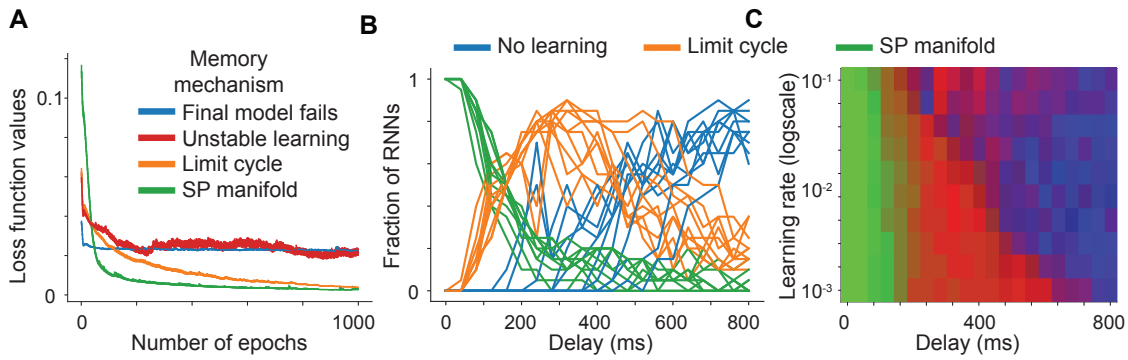

Figure 6.5. The observed scaling laws are robust to changes in the intrinsic neural time constant (τ).

Figure 7.5: This figure presents a control experiment where we doubled the neural time constant to $\tau = 20$ms and repeated the large-scale analysis from Figure 4. The results demonstrate that the fundamental learning dynamics and scaling laws are not dependent on this specific parameter.

(A) The characteristic loss curves for the four learning outcomes (successful convergence to a limit cycle or slow-point, and the two failure modes) are qualitatively identical to those observed with a faster time constant.

(B) The distribution of learned mechanisms across different learning rates and delay durations remains consistent with the previous findings, showing the same characteristic shift from slow-points to limit cycles as the delay increases.

(C) The empirical phase diagram for $\tau = 20$ms reveals the same dynamical phases and scaling boundaries as observed in the $\tau = 10$ms regime, confirming that the scaling laws are a general property of the learning problem, not an artifact of the specific neural timescale.

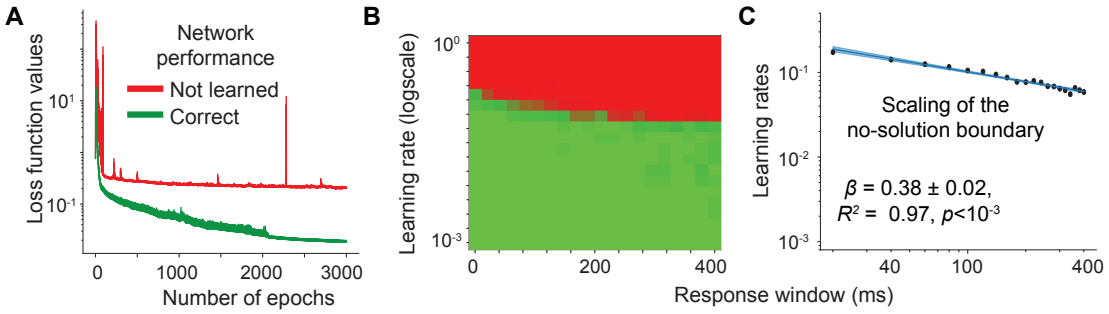

Figure 6.6. A control experiment demonstrates that the observed scaling laws are a direct consequence of the task's memory component.

Figure 7.6: To isolate the effect of memory, we removed the delay period (T_delay = 0) from the delayed cue-discrimination task and varied the duration of the response period (T_resp) instead.
(A) The loss curves for successfully trained ("correct") and unsuccessful ("not learned") networks exhibit typical training dynamics, similar to those in the main memory tasks.

(B) The empirical phase diagram for this "fixed-response" task. The x-axis now represents the duration of the required response. Notably, the boundary between the learned and not-learned regions lacks the clear, systematic structure observed in memory-dependent tasks (Figure 4.3).

(C) A power-law fit to the failure boundary, computed using the same methodology as in the main experiments, yields a critical exponent of $\beta = 0.38 \pm 0.02$. This near-zero slope confirms that the strong scaling behavior is eliminated when the memory requirement is removed from the task, highlighting that memory maintenance is the critical factor underlying these laws.