Structured flexibility in recurrent neural networks via neuromodulation

Julia C. Costacurta*
Stanford University
jcostac@stanford.edu

Shaunak Bhandarkar*
Stanford University
shaunakb@stanford.edu

David Zoltowski
Stanford University
dzoltow@stanford.edu

Scott W. Linderman
Stanford University
scott.linderman@stanford.edu

Abstract

A core aim in theoretical and systems neuroscience is to develop models that help us better understand biological intelligence. Such models range broadly in both complexity and biological plausibility. One widely-adopted example is task-optimized recurrent neural networks (RNNs), which have been used to generate hypotheses about how the brain's neural dynamics may organize to accomplish tasks. However, task-optimized RNNs typically have a fixed weight matrix representing the synaptic connectivity between neurons. From decades of neuroscience research, we know that synaptic weights are constantly changing, controlled in part by chemicals such as neuromodulators. In this work we explore the computational implications of synaptic gain scaling, a form of neuromodulation, using task-optimized low-rank RNNs. In our neuromodulated RNN (NM-RNN) model, a neuromodulatory subnetwork outputs a low-dimensional neuromodulatory signal that dynamically scales the low-rank recurrent weights of an output-generating RNN. In empirical experiments, we find that the structured flexibility in the NM-RNN allows it to both train and generalize with a higher degree of accuracy than low-rank RNNs on a set of canonical tasks. Additionally, via theoretical analyses we show how neuromodulatory gain scaling endows networks with gating mechanisms commonly found in artificial RNNs. We end by analyzing the low-rank dynamics of trained NM-RNNs, to show how task computations are distributed.

1 Introduction

Humans and animals show an innate ability to adapt and generalize their behavior across various environments and contexts. This suggests that the neural computations producing these behaviors must have flexible dynamics that are able to adjust to these novel conditions. Given the popularity of recurrent neural networks (RNNs) in studying such neural computations, a key question is whether this flexibility is (1) adequately and (2) accurately portrayed in RNN models of computation.

Traditional RNN models have fixed input, recurrent, and output weight matrices. Thus, the only way for an input to impact a dynamical computation is via the static input weight matrix. Prior work has shown that flexible, modular computation is possible with these models [1], but looking to the biology suggests alternative ways of modeling. In particular, experimental neuroscience research has shown that synaptic strengths in the brain (akin to weight matrix entries in RNNs) are constantly changing — in part due to the influence of neuromodulatory signals [2].

Neuromodulatory signals are powerful and prevalent influences on neural activity and subsequent behavior. Dopamine, a well-known example, is implicated in motor deficits resulting from Parkinson's disease and has been the subject of extensive study by neuroscientists [3]. For computational study,

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

neuromodulators are especially interesting because of their effects on synaptic connections and learning [4]. In particular, neuromodulators have been shown to alter synaptic strength between neurons, effectively reconfiguring circuit dynamics [5].

In this work, we seek to incorporate a neuromodulatory signal into task-trained RNN models. We propose a model consisting of a pair of RNNs: a small "neuromodulatory" RNN and a larger, low-rank "output-generating" RNN. The neuromodulatory RNN controls the weights of the output-generating RNN by scaling each rank-1 component of its recurrent weight matrix. This allows the network to produce flexible, yet structured dynamics that unfold over the course of a task. We first review background work in both the machine learning and computational neuroscience literature. Next, we introduce the model and provide some intuition for the impact of neuromodulation on the model's dynamics, relating our model to the canonical long short-term memory (LSTM) network. We end by presenting the performance and generalization capabilities of neuromodulated RNNs on a variety of neuroscience and machine learning tasks, showcasing the ability of a relatively simple, biologically-motivated augmentation to enhance the capacity of RNN models.

2 Background

First, we review related work in the theoretical neuroscience and machine learning literature.

2.1 Modeling neuromodulatory signals

Our work builds on a body of literature dating back to the 1980s, when pioneering computational neuroscientists added neuromodulation to small biophysical circuit models (for reviews, see [6–8]). These models consist of coupled differential equations whose biophysical parameters are carefully specified to simulate biologically-accurate spiking activity. As Marder relates in her retrospective review [5], such neuromodulatory models were created in response to neuronal circuit models which viewed circuit dynamics as "hard-wired". Neuromodulatory mechanisms offered an answer to experimental observations that anatomically fixed biological circuits were capable of producing variable outputs [9, 10]. Of particular relevance to this work, in his 1990 paper Abbott [11] showed that adding a neuromodulatory parameter to an ODE model of spiking activity allows a network of neurons to display capacity for both long- and short-term memory and gate the learning process, anticipating the LSTMs that would become prominent a few years later. We also draw comparisons between our model and the LSTM in the sections that follow. However, our work does not aim to model any specific biophysical system; rather, it aims to begin to bridge the gap between these highly biologically-accurate models and general network models (i.e. RNNs) of neuronal activity by adding a biologically-motivated form of structured flexibility.

More recent attempts to model neuromodulation have taken advantage of increased computational power. Prior work has investigated modulatory influence in spiking neural networks, for example by linearly scaling the firing rates of a subset of neurons [12], incorporating arousal-mediated modulatory signals to induce phase transitions [13], and facilitating credit assignment during learning [14]. Stroud et al. [15] use a balanced excitatory/inhibitory RNN to model motor cortex, and incorporate modulatory signals as constant multipliers on each neuron's activity. Our work similarly proposes a modulatory signal that impacts network activity, but we instead allow this signal to scale factors of the recurrence matrix. Duong et al. [16] use a similar factor scaling approach to model adaptive whitening in early sensory processing. Our approach differs by focusing on low-rank recurrence matrices in a task-oriented setting. In addition to RNN models, gain modulation has been explored in neural ODEs [17] and feedforward networks [18].

The works of Tsuda et al. [19] and Vecoven et al. [20] are most similar to what we present here. In Tsuda et al. [19], the authors train RNNs using constant, multiplicative neuromodulatory signals applied to pre-specified subsets of the recurrent weights. They show that these neuromodulatory signals allow an otherwise fixed network to perform variations of a task. In contrast, we employ time-dependent neuromodulatory signals that allow dynamics to evolve throughout tasks. Instead of pre-specifying the values and regions of impact of neuromodulators, we allow the model to learn the time-varying neuromodulatory signal and what segment of the neuronal population it impacts. Vecoven et al. [20] use a two-network approach in which a neuromodulatory network processes contextual information to alter the activation functions of a "main" feedforward deep neural network. They show that this "Neuro-Modulated Network" outperforms RNNs on meta-RL benchmarks. We also use a two-network approach in which one network modulates the parameters of the other, but instead applied to recurrent neural networks in a non-RL paradigm.

2.2 Hypernetworks

Our approach is closely related to recent work using hypernetworks to enhance model capacity. Ha et al. [21] use small networks (termed hypernetworks) to generate parameters for layers of larger networks. In their HyperRNN, a hypernetwork generates the weight matrix of an RNN as the linear combination of a learned set of matrices. We also allow our neuromodulatory network to specify the weight matrix of a larger RNN as a linear combination of a learned set of matrices; however, our learned matrices are rank-1 to facilitate easier dynamical analysis and faster training. It is also worth noting that in practice, Ha et al. [21] simplify their HyperRNN so that the hypernetwork scales the rows of a learned weight matrix, which could be seen as postsynaptic scaling in our model. Similarly, von Oswald et al. [22] study the ability of hypernetworks to learn in the multitask and continual learning setting. They find that hypernetworks trained to produce task-specific weight realizations achieve high performance on continual learning benchmarks. In exploring potential neuroscience applications of their work, they remark that while their approach might be unrealistic, a hypernetwork that outputs lower-dimensional modulatory signals could assist in implementing task-specific mode-switching. We seek to obtain similar performance gains with a biologically plausible model of neuromodulation.

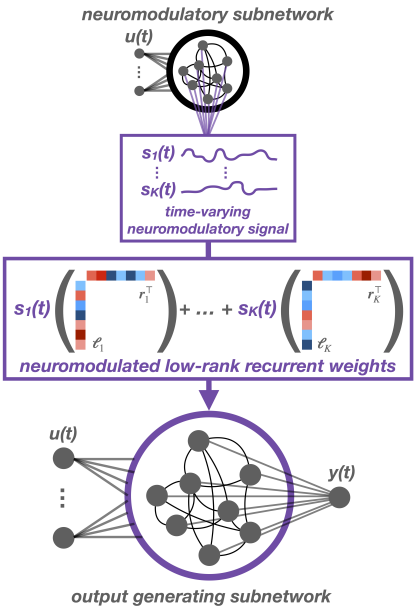

Figure 1: The NM-RNN consists of two subnetworks: a low-rank subnetwork that generates the output (bottom) and a smaller, full-rank neuromodulatory subnetwork (top).

2.3 Low-rank recurrent neural networks

For a variety of tasks of interest, measured neural recordings are often well-described by a set of lower dimensional latent variables [23, 24] (although see [25, 26] for an alternative view). Likewise, artificial neural networks trained to solve tasks that mimic those found in neural experiments also often exhibit low-rank structure [27]. Based on these findings, recurrent neural networks with low-rank weight matrices (also called low-rank RNNs, LR-RNNs) have emerged as a popular class of models for studying neural dynamics [28–30]. Importantly, low-rank RNNs are able to model nonlinear dynamics which evolve in a low-rank subspace, offering potential for visualization and interpretation. Here, we leverage low-rank RNNs as ideal candidates for neuromodulation, with each factor of the low-rank recurrence matrix becoming a possible target for synaptic scaling.

3 Neuromodulated recurrent neural networks

Motivated by the appealing dynamical structure of low-rank RNNs and the ability of neuromodulation to add structured flexibility, we propose the *neuromodulated RNN* (NM-RNN). The NM-RNN consists of two linked subnetworks corresponding to neuromodulation and output generation. The output generating subnetwork is a low-rank RNN, which admits a natural way to implement neuromodulation. We allow the output of the neuromodulatory subnetwork to scale the low-rank factors of the output generating subnetwork's weight matrix. In particular, we propose incorporating neuromodulatory drive via a coupled ODE with neuromodulatory subnetwork state $\boldsymbol{z}(t) \in \mathbb{R}^M$ and output-generating state $\boldsymbol{x}(t) \in \mathbb{R}^N$:

$$\tau_z \frac{\mathrm{d}\boldsymbol{z}(t)}{\mathrm{d}t} = -\boldsymbol{z}(t) + \boldsymbol{W}_z\, \phi(\boldsymbol{z}(t)) + \boldsymbol{B}_z \boldsymbol{u}(t) \tag{1}$$

$$\tau_x \frac{\mathrm{d}\boldsymbol{x}(t)}{\mathrm{d}t} = -\boldsymbol{x}(t) + \boldsymbol{W}_x\big(\boldsymbol{z}(t)\big)\, \phi(\boldsymbol{x}(t)) + \boldsymbol{B}_x \boldsymbol{u}(t) \tag{2}$$

$$\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{d}, \tag{3}$$

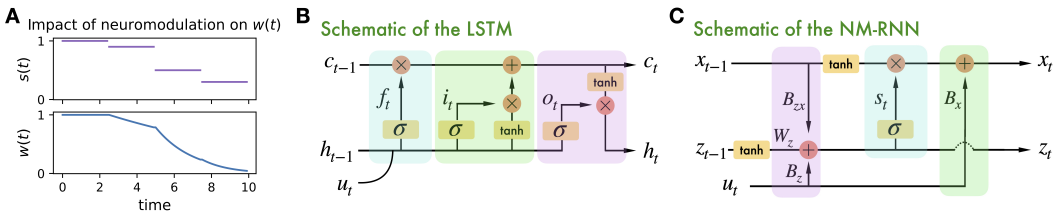

Figure 2: **A.** Illustration of how neuromodulatory signals $s(t)$ affect decay rates of the state $w(t)$ in a 1-D, simplified model (derived in Section 3.1), calculated with prespecified $s(t)$ signal. As $s(t)$ approaches 0, $w(t)$ decays more rapidly. **B. & C.** Visual comparison of an LSTM and an NM-RNN. Corresponding parts of the networks are highlighted with shaded rectangles. Blue: a forget gate computation. Green: an input gate to recurrent dynamics. Purple: recurrent feedback onto the modulatory state variable.

where the dynamics matrix $\boldsymbol{W}_x\big(\boldsymbol{z}(t)\big)$ is a function of the neuromodulatory subnetwork state $\boldsymbol{z}(t)$ via a neuromodulatory signal $\boldsymbol{s}(\boldsymbol{z}(t)) \in \mathbb{R}^K$ which scales each low-rank component of $\boldsymbol{W}_x$:

$$\boldsymbol{s}\big(\boldsymbol{z}(t)\big) = \sigma(\boldsymbol{A}_z \boldsymbol{z}(t) + \boldsymbol{b}_z) \qquad \boldsymbol{W}_x\big(\boldsymbol{z}(t)\big) = \sum_{k=1}^{K} s_k\big(\boldsymbol{z}(t)\big)\, \boldsymbol{\ell}_k \boldsymbol{r}_k^\top. \tag{4}$$

The output-generating subnetwork is modeled as a size-N low-rank RNN where $\boldsymbol{u}(t) \in \mathbb{R}^P$ are the inputs, $\boldsymbol{B}_x \in \mathbb{R}^{N \times P}$ are the input weights, and $\phi(\cdot)$ is the tanh nonlinearity. The neuromodulatory subnetwork is modeled as a small vanilla RNN with its own time-constant τ_z, recurrence weights $\boldsymbol{W}_z \in \mathbb{R}^{M \times M}$, and input weights $\boldsymbol{B}_z \in \mathbb{R}^{M \times P}$. To limit the capacity of the neuromodulatory subnetwork, we take its dimension M to be smaller than the output-generating subnetwork's dimension N. We set $\tau_z \gg \tau_x$ since neuromodulatory signals are believed to evolve relatively slowly [31]. The neuromodulatory subnetwork state $\boldsymbol{z}(t)$ alters the rank-K dynamics matrix $\boldsymbol{W}_x \in \mathbb{R}^{N \times N}$ via a K-dimensional linear readout $\boldsymbol{s}(\boldsymbol{z}(t))$, where $\sigma(\cdot)$ is the sigmoid nonlinearity. The components of $\boldsymbol{s}$ act as linear scaling factors on each rank-1 component $\boldsymbol{\ell}_k \boldsymbol{r}_k^\top \in \mathbb{R}^{M \times M}$ of $\boldsymbol{W}_x$. For ease of notation, in the rest of the text we write $\boldsymbol{s}(t)$ to mean $\boldsymbol{s}(\boldsymbol{z}(t))$. The output $\boldsymbol{y}(t) \in \mathbb{R}^O$ of the paired networks is a linear readout of $\boldsymbol{x}(t)$.

This augmentation to the RNN framework allows for structured flexibility in computation. In traditional RNNs, the recurrent weight matrix is fixed, and thus the inputs to the system can only perturb the state of the network. In the NM-RNN, the neuromodulatory subnetwork can use information from the inputs to dynamically up- and down-weight different low-rank components of the recurrent weight matrix, offering greater computational flexibility. As we will see below, this flexibility also allows the network to reuse dynamical components across different tasks and task conditions.

3.1 Mathematical intuition

To gain some intuition for the potential impacts of neuromodulation on RNN dynamics, first consider the case where $\boldsymbol{W}_x$ is symmetric (i.e., $\boldsymbol{\ell}_k = \boldsymbol{r}_k \, \forall k$), where $\{\boldsymbol{\ell}_k\}_{1 \le k \le K}$ forms an orthonormal set, where the nonlinearity is removed (i.e., $\phi(x) = x$), and where there are no inputs (i.e., $\boldsymbol{u}(t) = 0 \, \forall t$). We can then reparameterize the system with a new hidden state $\boldsymbol{w}(t) = \boldsymbol{L}^\top \boldsymbol{x}(t)$, where $\boldsymbol{L} \in \mathbb{R}^{N \times K}$ is the matrix whose columns are $\boldsymbol{\ell}_k$ (so that $\boldsymbol{L}^T \boldsymbol{L} = I$). This produces decoupled dynamics:

$$\tau_x \frac{\mathrm{d}\boldsymbol{w}(t)}{\mathrm{d}t} = -\boldsymbol{w}(t) + \mathbf{S}(t)\boldsymbol{w}(t) \tag{5}$$

where $\boldsymbol{S}(t) = \mathrm{diag}\,(\boldsymbol{s}(t))$. Solving this ODE gives an equation for the components of $\boldsymbol{w}(t)$:

$$w_k(t) = w_k(0)\, \exp\left(- \int_0^t \frac{(1 - s_k(t'))}{\tau_x} \, \mathrm{d}t' \right)$$

From this equation and the visualization in Fig. 2A, we see that the decay rate of each component $w_k(t)$ is governed by its corresponding neuromodulatory signal $s_k(t)$. In this way, $\boldsymbol{s}(t)$ can effectively speed up or slow down decay of dynamic modes, similar to gating in an LSTM.

3.2 Connection to LSTMs

Having observed that neuromodulation can alter the timescales of dynamics, note further that the low-rank update for the linearized NM-RNN in eq. (5) mirrors the cell-state update equation for a long short-term memory (LSTM) cell [32]. Specifically, the neuromodulatory signal $\boldsymbol{s}(t)$ resembles the forget gate of an LSTM (fig. 2B). Indeed, as in eq. (5), if we linearize the output-generating subnetwork of the NM-RNN and assume that $\boldsymbol{L} = \boldsymbol{R}$ (so that $\boldsymbol{W}_x$ is symmetric) and $\boldsymbol{L}^T\boldsymbol{L} = I$, then for $\boldsymbol{w}(t) = \boldsymbol{L}^T\boldsymbol{x}(t)$ and $\tau_x = 1$, the discretized low-rank dynamics are given by

$$\boldsymbol{w}_t = \boldsymbol{s}_t \odot \boldsymbol{w}_{t-1} + \boldsymbol{L}^T\boldsymbol{B}_x\boldsymbol{u}_t \tag{6}$$

LSTMs have two states that recurrently update across each timestep t: a hidden state $\boldsymbol{h}_t \in \mathbb{R}^{N_{\text{LSTM}}}$ and a cell state $\boldsymbol{c}_t \in \mathbb{R}^{N_{\text{LSTM}}}$. Equation (6) closely mirrors the cell-state update of the LSTM:

$$\boldsymbol{c}_t = \mathbf{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t \tag{7}$$

Here, the forget gate $\mathbf{f}_t$ is a form of modulation that depends on the LSTM hidden state $\boldsymbol{h}_t$, much like the NM-RNN's neuromodulatory signal $\boldsymbol{s}(t)$ is a form of modulation depending on the NM-RNN's neuromodulatory subnetwork state $\boldsymbol{z}(t)$. The second term $\boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t$ can be viewed as a gated transformation of the input signal $\boldsymbol{u}(t)$. In fact, under suitable assumptions, we show that the dynamics of an NM-RNN can be reproduced by those of an LSTM (see Supplementary Material, Proposition 1).

As a gated analog of the RNN, the LSTM has enjoyed greater success than ordinary RNNs in performing tasks that involve keeping track of long-distance dependencies in the input signal [33]. Thus, highlighting the connection between the NM-RNN and LSTM suggests the NM-RNN's ability to successfully model long-timescale dependencies, unlike regular RNNs (see Section 6).

4 Time interval reproduction

To evaluate the potential of neuromodulated RNNs to add structured flexibility, we first consider a timing task since neuromodulators such as dopamine are implicated in time measurement and perception [34]. In the Measure-Wait-Go (MWG) task (Fig. 3A) [35], the network receives a 3-channel input containing the measure, wait, and go cues. The network must measure the interval between the measure and wait cues, and reproduce it at the go cue by outputting a linear ramp of the same duration. Tasks such as this one are commonly used to study the neural underpinnings of timing perception in humans and non-human primates [36, 37].

4.1 Experiment matches theory for rank-1 networks

To investigate how the NM-RNN's neuromodulatory signal is constrained by the task requirements, we first consider a class of analytically tractable NM-RNNs: networks for which the output-generating subnetwork is linear (i.e., the tanh nonlinearity is replaced with the identity function) and rank-1. In this case, there is one pair of row and column factors, $\boldsymbol{\ell}$ and $\boldsymbol{r}$, respectively. If the target output signal is given by $f(t)$ and there are no inputs, then an NM-RNN that successfully produces this output signal will precisely have the neuromodulatory signal,

$$\boldsymbol{s}(t) = \frac{f(t) + \tau_x f'(t) - \boldsymbol{d}}{(\boldsymbol{\ell}^\top \boldsymbol{r})\left(f(t) - \boldsymbol{c}^T\boldsymbol{w}^\perp(0)e^{-t/\tau_x} - \boldsymbol{d}\right) + (\boldsymbol{c}^\top\boldsymbol{\ell})\left(\boldsymbol{r}^\top\boldsymbol{w}^\perp(0)e^{-t/\tau_x}\right)},$$

where our readout is $y = \boldsymbol{c}^\top\boldsymbol{x} + \boldsymbol{d}$ and $\boldsymbol{w}^\perp(t) := \boldsymbol{x}(t) - \frac{1}{||\boldsymbol{\ell}||_2^2}\boldsymbol{\ell}\boldsymbol{\ell}^\top\boldsymbol{x}(t)$ is the component of $\boldsymbol{x}(t)$ evolving outside of the column space of $\boldsymbol{\ell}$ (viewed as a matrix in $\mathbb{R}^{N\times 1}$). For the full derivation, see the Supplementary Material, Section B. If we assume further that $\boldsymbol{w}^\perp(0)$ is sufficiently small and τ_x is also sufficiently small, then we may make the approximation,

$$\boldsymbol{s}(t) \approx \frac{f(t) + \tau_x f'(t) - \boldsymbol{d}}{\boldsymbol{\ell}^\top\boldsymbol{r}\,(f(t) - \boldsymbol{d})} = \frac{1}{\boldsymbol{\ell}^T\boldsymbol{r}}\left(1 + \frac{\tau_x f'(t)}{f(t) - \boldsymbol{d}}\right). \tag{8}$$

In the MWG task, no inputs are provided to the network during the ramping period following the go cue, so eq. (8) applies. In fig. 3B, we show that the neuromodulatory signal of a trained rank-1 NM-RNN during the ramping period matches closely with the theoretical prediction made by eq. (8) for both trained and extrapolated target intervals.

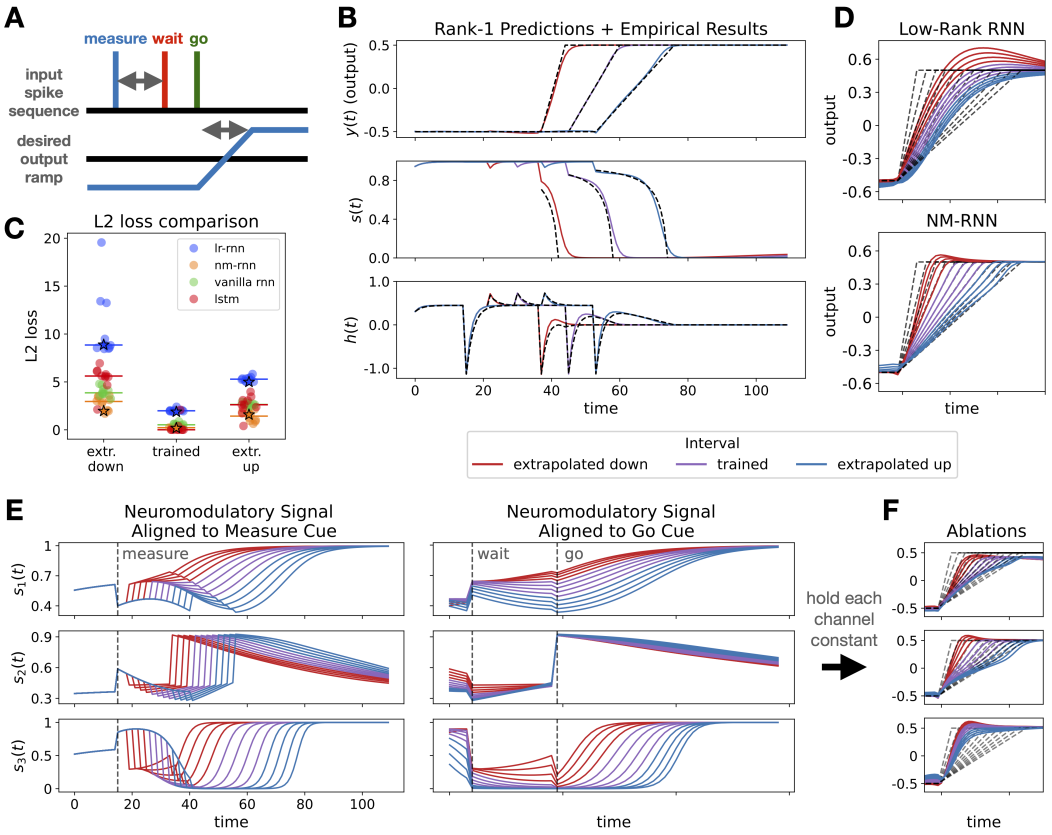

Figure 3: **A.** Visualization of Measure-Wait-Go task. **B.** Theoretical predictions (dashed lines) match closely with empirical results (solid lines) for rank-1 network. **C.** L2 loss comparsion of four model types on MWG task. We trained 10 randomly initialized parameter-matched low-rank RNNs, NM-RNNs, vanilla RNNs, and LSTMs. Median losses shown by bars. Performance of visualized LR-RNN and NM-RNN are starred (example visualizations for all four models shown in Supp. Fig. 1). **D.** Comparison of model-generated output ramps for both trained (purple) and extrapolated (red, blue) intervals. Dashed lines show target outputs. **E.**Three-dimensional neuromodulatory signal $\boldsymbol{s}(t)$ for trained/extrapolated intervals. Left, traces are aligned to start of trial. Right, traces are aligned to 'go' cue. **F.** Resulting output traces when ablating each component of $\boldsymbol{s}(t)$. In all panels, colors reflect trained/extrapolated intervals (see legend). For output plots, dashed grey lines are targets. Additional model visualizations in Supp. Fig. 1-2.

4.2 Improved generalization and interpretability on timing task for rank-3 networks

To continue our analysis of NM-RNNs on the MWG task, we increase the rank of the output-generating subnetwork to three. We do this to compare to the networks shown in Beiran et al. [35] and to showcase networks with more degrees of structured flexibility. In Beiran et al. [35], the authors show that rank-3 low-rank RNNs perform and extrapolate better on this task when provided a tonic context-dependent input, which varies depending on the length of the desired interval. As we have mentioned, such sensory inputs to the network may only alter the resulting dynamics by being passed through the input weight matrix. We propose the NM-RNN as an alternative mechanism by which inputs may alter the network dynamics.

We trained parameter-matched NM-RNNs ($N = 100,\ M = 5,\ K = 3,\ \tau_x = 10,\ \tau_z = 100$), LR-RNNs ($N = 106,\ K = 3,\ \tau = 10$), vanilla RNNs ($N = 31,\ \tau = 10$), and LSTMs ($N = 15$) to reproduce four intervals, then tested their extrapolation to longer and shorter intervals. In the LR-RNN and NM-RNN, the low-rank matrix was chosen to have rank 3, as in Beiran et al. [35]. In fig. 3C, we plot the L2 losses for ten instances of each model. We see that although the vanilla RNN, LSTM, and NM-RNN are all able to train accurately, the NM-RNN consistently achieves a lower loss on the extrapolated intervals. In fig. 3D, we then show outputs for a typical LR-RNN and NM-RNN

(performance of these visualized networks indicated by stars in fig. 3C, visualizations of vanilla RNN and LSTM shown in Supp. Fig. 1). The outputs of the NM-RNN have more accurate slope and shape for both trained and extrapolated intervals, with the LR-RNN failing to reproduce shorter and longer extrapolated intervals.

We then investigate how the neuromodulatory signal $\boldsymbol{s}(t)$ contributes to the task computation. Figure 3E shows the three dimensions of $\boldsymbol{s}(t)$ plotted over the full range of trained and extrapolated intervals. Each dimension shows activity correlated to particular stages of the task. In fig. 3E (right), we see that $s_1(t)$ and $s_3(t)$ have activity highly correlated to the measured interval. In particular, we can see that between the wait and go cues, $s_1(t)$ separates shorter intervals from longer ones, setting up initial dynamics for the go period when the ramp is generated. The third component $s_3(t)$ appears to be involved with ending the output ramp, since it saturates first for the shorter intervals and then the longer ones. Figure 3F shows the result of ablating each dimension of $\boldsymbol{s}(t)$ by keeping that component fixed around its initial value. We see that performance suffers in all cases, especially when ablating the effect of $s_1(t)$ and $s_3(t)$. Most dramatically, ablating $s_3(t)$ destroys the ability of the network to change the slope of the output ramp appropriately. These results show that the network uses its neuromodulatory signal to process timing information and generalize across task conditions.

5 Reusing dynamics for multitask learning

Next, we move beyond generalization within a single task to investigate the capabilites of the NM-RNN when switching between tasks. There has been recent interest in studying how neural network models might reassemble learned dynamical motifs to accomplish multiple tasks [1, 38]. Driscoll et al [1] showed that an RNN trained to perform an array of tasks shares modular dynamical motifs across task periods and between tasks. With this result in mind, we were curious how the NM-RNN might use its neuromodulatory signal to flexibly reconfigure dynamics across tasks.

We performed our analysis using the four-task set from Duncker et al. [39], which includes the tasks DelayPro, DelayAnti, MemoryPro, and MemoryAnti illustrated in fig. 4A. In the DelayPro task, the network receives a three-channel input consisting of a fixation input and two sensory inputs which encode an angle $\theta \in [0, 2\pi)$ as $(\sin(\theta), \cos(\theta))$. The fixation input starts and remains at 1, then drops to 0 to signal the start of the *readout* period, when the network must generate its response. The sensory inputs appear after a delay, and persist throughout the trial. The goal of the network is to produce a three-channel output which reproduces the fixation and sensory inputs. In the MemoryPro task, the sensory inputs disappear before the readout period, requiring the network to store θ. In the 'Anti' tasks, the networks must instead produce the opposite sensory outputs, $(\sin(\theta+\pi), \cos(\theta+\pi))$, during the readout period. The task context is fed in as an additional one-hot input. These tasks are analogous to variants of the center-out reaching task, which has been used to study the neural mechanisms of motion in non-human primates [40].

To study the potential of NM-RNNs to flexibly reconfigure dynamics to perform a new task, we only fed the contextual inputs to the neuromodulatory subnetwork, and not to the output-generating subnetwork. This required the model to reuse the output-generating subnetwork's weights when adding a new task. We trained an NM-RNN to perform the first three tasks in the set (DelayPro, DelayAnti, MemoryPro), then froze the weights of the output-generating subnetwork and retrained only the neuromodulatory subnetwork's weights on the fourth task, MemoryAnti. We compared this to retraining the input weights of LR-RNNs, vanilla RNNs, and LSTMs, to investigate two strategies of processing context.

We trained parameter-matched NM-RNNs ($N = 100,\ M = 20,\ K = 3,\ \tau_x = 10,\ \tau_z = 100$), LR-RNNs ($N = 100,\ K = 3,\ \tau = 10$), vanilla RNNs ($N = 18,\ \tau = 10$), and LSTMs ($N = 8$) in this training/retraining framework. Figure 4B shows performance of example networks on the trained and retrained tasks, using the percent correct metric from Driscoll et al. [1]. The NM-RNN matches the performance of the LSTM and higher-rank vanilla RNN, and considerably outperforms the LR-RNN with no modulation. This performance gain over LR-RNNs is not the result of retraining more parameters; in fact, due to the contrasting sizes of the neuromodulatory and low-rank subnetworks, the input weight matrix of the comparison LR-RNN contains more parameters than the entire neuromodulatory subnetwork, since it must process all inputs (context, sensory, and fixation). To see exactly how the recurrent dynamics were rearranged for this new task, we plotted the neuromodulatory signal of an example network for learned and extrapolated tasks in fig. 4C.

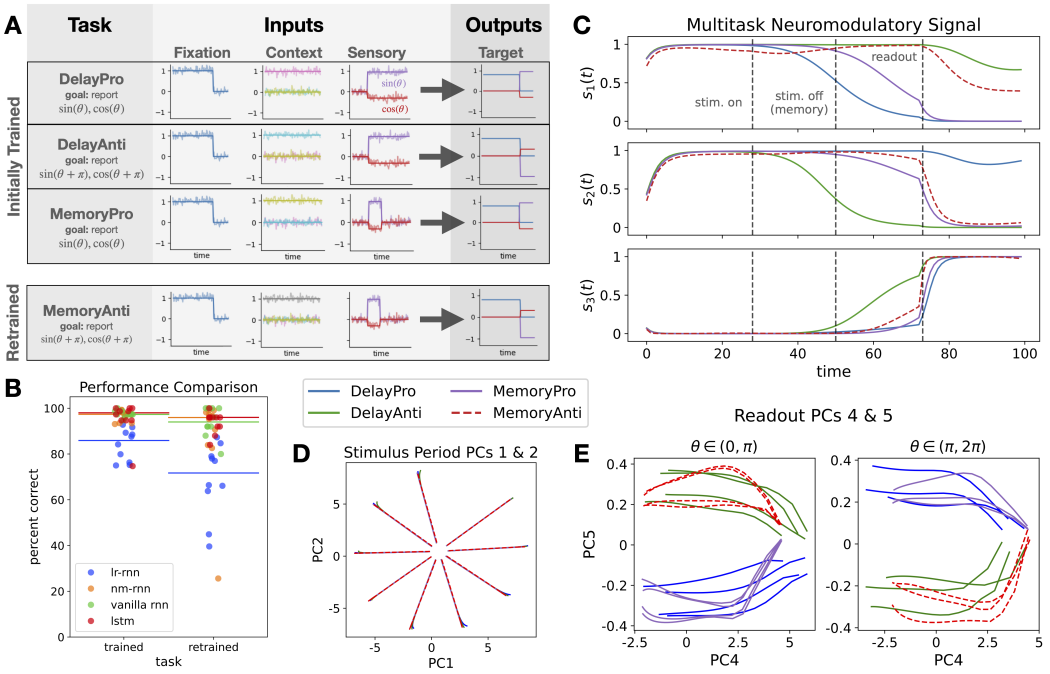

Figure 4: **A.** Depiction of inputs and targets for four tasks. Networks were trained on the first three tasks, then context-processing weights were retrained on the fourth task (final row). **B.** Performance comparison for four model types in multitask setting, for three initially trained tasks and retrained MemoryAnti task. Median performance shown by bars, with color of bar split in case of overlapping. We trained 10 randomly initialized parameter-matched low-rank RNNs, NM-RNNs, vanilla RNNs, and LSTMs. **C.** Neuromodulatory signal for an example network. **D.** Dynamical analysis of network activity during different stages of the tasks. (Left) PCs 1&2 during the stimulus period show a ring attractor which stores the measured angle. (Right) Sign of PC5 during readout period corresponds to Pro/Anti. Additional model visualizations in Supp. Fig. 3-4.

We then analyzed the dynamical structure of one of the NM-RNNs by performing PCA on the output-generating subnetwork's state $\boldsymbol{x}(t)$ for a variety of input angles. Figure 4D shows the first two PCs of the neural activity during the stimulus presentation period (before the stimulus shut off for Memory trials). During this period, the neural activity spreads out to arrange on a ring according to the measured angle. After the stimulus disappears in the MemoryPro/Anti tasks, the neural activity decays back along these axes, but it is still decodable based on its angle from the origin (see Supp. Fig. 4). To find how this model encoded Pro/Anti versions of tasks, we performed another PCA on the neural activity during the readout period. As shown in fig. 4E, the sign of PC5 during this period is correlated with whether the task is Pro or Anti. Curiously, the positive/negative relationship flips for $\theta \in (\pi, 2\pi)$, likely relating to the symmetric structure of sine and cosine. These results show the ability of the NM-RNN to flexibly reconfigure the dynamics of the output-generating subnetwork, both to solve multiple tasks simultaneously and to generalize to a novel task.

6 Capturing long-term dependencies via neuromodulation

Inspired by the similarity between the coupled NM-RNN and the LSTM (see section 3.2), we designed a sequence-related task with long-term dependencies, called the *Element Finder Task* (EFT). On this task, gated models like the NM-RNN outperform ordinary RNNs. When endowed with suitable feedback coupling from the output-generating subnetwork to the neuromodulatory subnetwork, the NM-RNN demonstrates LSTM-like performance on the EFT, while vanilla RNNs (with matched parameter count) fail to solve this task.

In the EFT (fig. 5A), the input stream consists of a query index, $q \in \{0, 1, \ldots, T-1\}$ followed by a sequence of T randomly chosen integers. The goal of the model is to recover the value of the element at index q from the sequence of integers. At each time for $t \geq 1$, the tth element of the sequence is

passed as a one-dimensional input to the model. At time $t = T$, the model must output the value of the element at index q in the sequence. For our results (shown below), we took $T = 25$.

We trained several NM-RNNs, LR-RNNs, full-rank RNNs, and LSTMs on the EFT, conserving the total parameter count across networks. To emphasize its connection to the LSTM, each NM-RNN included an additional feedback coupling from $\boldsymbol{x}(t)$ to $\boldsymbol{z}(t)$:

$$\tau_z \frac{\mathrm{d}\boldsymbol{z}(t)}{\mathrm{d}t} = -\boldsymbol{z}(t) + \boldsymbol{W}_z \phi(\boldsymbol{z}(t)) + (\boldsymbol{B}_{zx}\, \phi(\boldsymbol{x}(t)) + \boldsymbol{b}_{zx}) + \boldsymbol{B}_z \boldsymbol{u}(t) \tag{9}$$

Each model used a linear readout with no readout bias. The resulting performances of each model tested are shown in fig. 5B. Figure 5C moreover illustrates the learning dynamics (as measured by MSE loss) for a single run of selected networks. Like LSTMs, NM-RNNs successfully perform the task, whereas low- and full-rank RNNs largely fail to do so.

To understand how a particular NM-RNN ($N = 18, M = 5, K = 8, \tau_x = 2, \tau_z = 10$) uses neuromodulatory gating to solve the EFT, we visualize the trial-averaged behavior of different components of $\boldsymbol{s}(t)$ across query indices ($q = 5, 10, 15, 20$), revealing that certain components of $\boldsymbol{s}(t)$ transition between 0 and 1 on a timescale correlated to the query index q (fig. 5D; left and right); while other components zero out (fig. 5; middle). Visualizing a low-dimensional projection of $\boldsymbol{z}(t)$ across different query indices reveals that $\boldsymbol{z}(t)$ settles to a fixed point on an approximate line attractor encoding query index q (fig. 5E). These findings show that $\boldsymbol{z}(t)$ attends to the query index, facilitating gate-switching behavior in $\boldsymbol{s}(t)$ upon arrival of the queried element.

Next, we analyze $\boldsymbol{x}(t)$ by visualizing its top 2 principal components across each combination of the query indices $q = 5, 10$, and 15 and the target element values $-10, -5, 0, 5$, and 10 (fig. 5F). Trajectories with different element values but the same query index start at the same location. Each trajectory converges towards the origin, and upon arrival of the query timestep, rapidly moves to a fixed point on an approximate line attractor that encodes element value. The arrangement of fixed points along this line moreover preserves the ordering of their corresponding element values. In summary, these results show that the NM-RNN solves the EFT by distributing its computations across the neuromodulatory subnetwork, which attends to the query index, and the output-generating subnetwork, which retrieves the target element value.

7 Discussion

As we have shown, neuromodulated RNNs display an increased ability to both perform and generalize on tasks, demonstrating an important computational implication of synaptic gain scaling. This enhanced performance is enabled by the structured flexibility neuromodulation adds to the dynamics of the network, via modulation of the singular values of the low-rank recurrent weight matrix. As we have shown both theoretically and in practice, this flexibility lends itself particularly well to tasks with timing-related variability. In addition, we saw performance gains over the LR-RNN for the multitask paradigm. Curiously, the gating-like dynamics introduced by adding neuromodulation are reminiscent of the canonical LSTM, and we can prove equivalence under certain conditions.

Limitations. One limitation of this work relates to the scale of the networks tested. Our networks were on the scale of $N \approx 100$ neurons at their largest, as opposed to other related works which use neuron counts in the thousands. However, we found that this number of neurons was adequate to perform the tasks we presented. We also have yet to compare our results to neural data, limiting our ability to draw biological conclusions.

Future Work. We are excited at the potential of future work to further bridge the gap between biophysical and recurrent neural network models. To expand on the NM-RNN model, we aim to embrace the broad range of roles neuromodulation can play in neural circuits. Potential future avenues include: (1) sparsifying the rank-1 components of the recurrent weight matrix to better imitate the ability of neuromodulators to act on spatially localized subpopulations of cells; (2) changing the readout function of $\boldsymbol{s}(t)$ to enable it to take both negative and positive values, in line with the ability of neuromodulators to act as both excitatory and inhibitory; and (3) investigating how different neuromodulatory effects may act on different timescales, both during task completion and learning over longer timescales [4, 5]. More generally, each neuron (or synapse) could have an internal state beyond its firing rate which is manipulated by neuromodulators, as in recent work investigating the role of modulation in generating novel dynamical patterns [41, 42]. Beyond neuromodulators, there

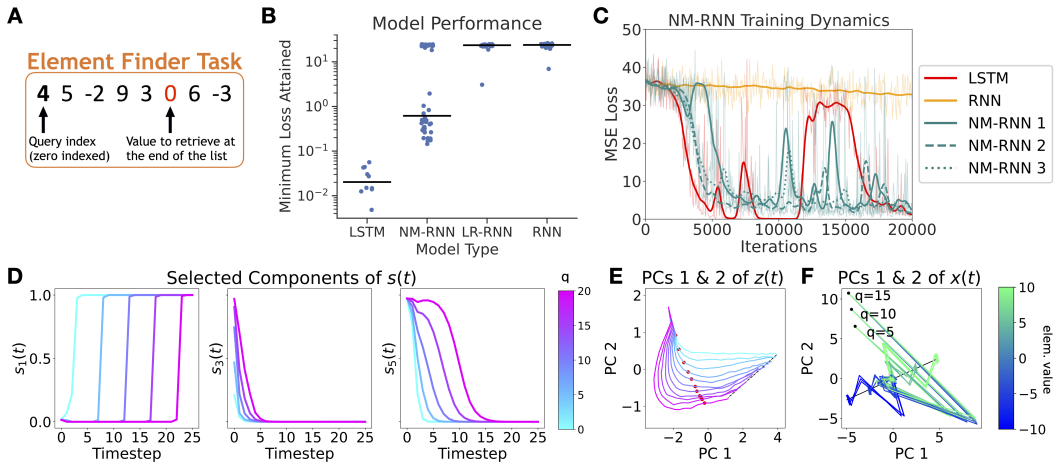

Figure 5: **A.** Visualization of the Element Finder Task. **B.** MSE losses attained across multiple runs in different classes of models trained on the EFT (median is indicated by black lines). **C.** Training loss curves for selected parameter-matched models. The NM-RNNs have hyperparameter counts 1: (M=5, N=18, R=8), 2: (M=5, N=13, R=12), and 3: (M=10, N=12, R=7). **D.** Visualization of selected components of $\boldsymbol{s}(t)$ for an example NM-RNN, shown across different query indices. **E.** Trajectories for the top two PCs of $\boldsymbol{z}(t)$ across different query indices. The different trajectories converge to an approximate line attractor (black) encoding query index. The time at which the queried element arrives is marked in red. **F.** Top two PCs of $\boldsymbol{x}(t)$, visualized for different query indices and target element values. Each trajectory converges to a fixed point on an approximate line attractor encoding element value. Each curve shown in **D**, **E**, and **F** is averaged over 100 trials. Additional visualizations in Supp. Fig. 5-6.

exist a multitude of extrasynaptic signaling mechanisms in the brain, such as neuropeptides and hormones, each with their own computational and modeling implications.

In this work, we only analyzed networks post-training. We are also curious how our computational mechanism of neuromodulation impacts the network during learning. Prior work has modeled the role of neuromodulation in learning, for example, by augmenting the traditional Hebbian learning rule with neuromodulation to implement a three-factor learning rule [43], and by using neuromodulation to create a more biologically plausible learning rule for RNNs [44]. Our neuromodulatory signal could induce similar mechanisms during learning.

Acknowledgements and Disclosure of Funding

We thank Laura Driscoll, Lea Duncker, Gyu Heo, Bernardo Sabatini, and the members of the Linderman Lab for helpful feedback throughout this project. This work was supported by grants from the NIH BRAIN Initiative (U19NS113201, R01NS131987, & RF1MH133778) and the NSF/NIH CRCNS Program (R01NS130789). J.C.C. is funded by the NSF Graduate Research Fellowship, Stanford Graduate Fellowship, and Stanford Diversifying Academia, Recruiting Excellence (DARE) Fellowship. D.M.Z. is funded by the Wu Tsai Interdisciplinary Postdoctoral Research Fellowship. S.W.L. is supported by fellowships from the Simons Collaboration on the Global Brain, the Alfred P. Sloan Foundation, and the McKnight Foundation. The authors have no competing interests to declare.

References

[1] Laura N Driscoll, Krishna Shenoy, and David Sussillo. Flexible multitask computation in recurrent networks utilizes shared dynamical motifs. *Nature Neuroscience*, 27(7):1349–1363, 2024.

[2] Ami Citri and Robert C Malenka. Synaptic plasticity: multiple forms, functions, and mechanisms. *Neuropsychopharmacology*, 33(1):18–41, 2008.

[3] Marjan Jahanshahi, Catherine RG Jones, Jan Zijlmans, Regina Katzenschlager, Lucy Lee, Niall Quinn, Chris D Frith, and Andrew J Lees. Dopaminergic modulation of striato-frontal connectivity during motor timing in parkinson's disease. *Brain*, 133(3):727–745, 2010.

[4] Peter Dayan. Twenty-five lessons from computational neuromodulation. *Neuron*, 76(1):240–256, 2012.

[5] Eve Marder. Neuromodulation of neuronal circuits: back to the future. *Neuron*, 76(1):1–11, 2012.

[6] Jean-Marc Fellous and Christiane Linster. Computational models of neuromodulation. *Neural Computation*, 10(4):771–805, 1998.

[7] Ronald M Harris-Warrick and Eve Marder. Modulation of neural networks for behavior. *Annual Review of Neuroscience*, 14(1):39–57, 1991.

[8] Alex H Williams, Albert W Hamood, and Eve Marder. Neuromodulation in small networks. In *Encyclopedia of Computational Neuroscience*, pages 2300–2313. Springer, 2022.

[9] Eve Marder. Mechanisms underlying neurotransmitter modulation of a neuronal circuit. *Trends in Neurosciences*, 7(2):48–53, 1984.

[10] Eve Marder and Scott L Hooper. Neurotransmitter modulation of the stomatogastric ganglion of decapod crustaceans. In *Model Neural Networks and Behavior*, pages 319–337. Springer, 1985.

[11] Larry F Abbott. Modulation of function and gated learning in a network memory. *Proceedings of the National Academy of Sciences*, 87(23):9241–9245, 1990.

[12] Ann Kennedy, Prabhat S Kunwar, Ling-yun Li, Stefanos Stagkourakis, Daniel A Wagenaar, and David J Anderson. Stimulus-specific hypothalamic encoding of a persistent defensive state. *Nature*, 586(7831): 730–734, 2020.

[13] Lia Papadopoulos, Suhyun Jo, Kevin Zumwalt, Michael Wehr, David A McCormick, and Luca Mazzucato. Modulation of metastable ensemble dynamics explains optimal coding at moderate arousal in auditory cortex. *bioRxiv*, April 2024.

[14] Yuhan Helena Liu, Stephen Smith, Stefan Mihalas, Eric Shea-Brown, and Uygar Sümbül. Cell-type–specific neuromodulation guides synaptic credit assignment in a spiking neural network. *Proceedings of the National Academy of Sciences*, 118(51):e2111821118, 2021.

[15] Jake P Stroud, Mason A Porter, Guillaume Hennequin, and Tim P Vogels. Motor primitives in space and time via targeted gain modulation in cortical networks. *Nature Neuroscience*, 21(12):1774–1783, December 2018.

[16] Lyndon Duong, David Lipshutz, David Heeger, Dmitri Chklovskii, and Eero P Simoncelli. Adaptive whitening in neural populations with gain-modulating interneurons. In *International Conference on Machine Learning*, pages 8902–8921. PMLR, 2023.

[17] Ilze Amanda Auzina, Çağatay Yıldız, Sara Magliacane, Matthias Bethge, and Efstratios Gavves. Modulated neural odes. *Advances in Neural Information Processing Systems*, 36:44572–44594, 2023.

[18] Laura B Naumann, Joram Keijser, and Henning Sprekeler. Invariant neural subspaces maintained by feedback modulation. *Elife*, 11:e76096, 2022.

[19] Ben Tsuda, Stefan C Pate, Kay M Tye, Hava T Siegelmann, and Terrence J Sejnowski. Neuromodulators generate multiple context-relevant behaviors in a recurrent neural network by shifting activity hypertubes. *bioRxiv*, 2021.

[20] Nicolas Vecoven, Damien Ernst, Antoine Wehenkel, and Guillaume Drion. Introducing neuromodulation in deep neural networks to learn adaptive behaviours. *PloS one*, 15(1):e0227922, 2020.

[21] David Ha, Andrew M. Dai, and Quoc V. Le. Hypernetworks. In *International Conference on Learning Representations*, 2017.

[22] Johannes von Oswald, Christian Henning, João Sacramento, and Benjamin F. Grewe. Continual learning with hypernetworks. In *International Conference on Learning Representations*, 2020.

[23] John P Cunningham and Byron M Yu. Dimensionality reduction for large-scale neural recordings. *Nature Neuroscience*, 17(11):1500–1509, 2014.

[24] Saurabh Vyas, Matthew D Golub, David Sussillo, and Krishna V Shenoy. Computation through neural population dynamics. *Annual Review of Neuroscience*, 43:249–275, 2020.

[25] Carsen Stringer, Marius Pachitariu, Nicholas Steinmetz, Matteo Carandini, and Kenneth D Harris. High-dimensional geometry of population responses in visual cortex. *Nature*, 571(7765):361–365, 2019.

[26] Dean A Pospisil and Jonathan W Pillow. Revisiting the high-dimensional geometry of population responses in visual cortex. *bioRxiv*, 2024.

[27] Friedrich Schuessler, Francesca Mastrogiuseppe, Alexis Dubreuil, Srdjan Ostojic, and Omri Barak. The interplay between randomness and structure during learning in rnns. *Advances in Neural Information Processing Systems*, 33:13352–13362, 2020.

[28] Francesca Mastrogiuseppe and Srdjan Ostojic. Linking connectivity, dynamics, and computations in low-rank recurrent neural networks. *Neuron*, 99(3):609–623.e29, August 2018.

[29] Alexis Dubreuil, Adrian Valente, Manuel Beiran, Francesca Mastrogiuseppe, and Srdjan Ostojic. The role of population structure in computations through neural dynamics. *Nature Neuroscience*, 25(6):783–794, 2022.

[30] Manuel Beiran, Alexis Dubreuil, Adrian Valente, Francesca Mastrogiuseppe, and Srdjan Ostojic. Shaping dynamics with multiple populations in low-rank recurrent networks. *Neural Computation*, 33(6):1572–1615, 05 2021.

[31] Eve Marder and Ronald L Calabrese. Principles of rhythmic motor pattern generation. *Physiological Reviews*, 76(3):687–717, 1996.

[32] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, November 1997.

[33] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. *Neural Computation*, 31(7):1235–1270, 07 2019.

[34] Allison E Hamilos, Giulia Spedicato, Ye Hong, Fangmiao Sun, Yulong Li, and John A Assad. Slowly evolving dopaminergic activity modulates the moment-to-moment probability of reward-related self-timed movements. *Elife*, 10, December 2021.

[35] Manuel Beiran, Nicolas Meirhaeghe, Hansem Sohn, Mehrdad Jazayeri, and Srdjan Ostojic. Parametric control of flexible timing through low-dimensional neural manifolds. *Neuron*, 111(5):739–753, 2023.

[36] Wilbert Zarco, Hugo Merchant, Luis Prado, and Juan Carlos Mendez. Subsecond timing in primates: comparison of interval production between human subjects and rhesus monkeys. *Journal of Neurophysiology*, 102(6):3191–3202, 2009.

[37] Akihisa Mita, Hajime Mushiake, Keisetsu Shima, Yoshiya Matsuzaka, and Jun Tanji. Interval time coding by neurons in the presupplementary and supplementary motor areas. *Nature Neuroscience*, 12(4):502–507, 2009.

[38] Guangyu Robert Yang, Madhura R Joglekar, H Francis Song, William T Newsome, and Xiao-Jing Wang. Task representations in neural networks trained to perform many cognitive tasks. *Nature Neuroscience*, 22 (2):297–306, 2019.

[39] Lea Duncker, Laura Driscoll, Krishna V Shenoy, Maneesh Sahani, and David Sussillo. Organizing recurrent network dynamics by task-computation to enable continual learning. *Advances in Neural Information Processing Systems*, 33:14387–14397, 2020.

[40] Cynthia A Chestek, Aaron P Batista, Gopal Santhanam, M Yu Byron, Afsheen Afshar, John P Cunningham, Vikash Gilja, Stephen I Ryu, Mark M Churchland, and Krishna V Shenoy. Single-neuron stability during repeated reaching in macaque premotor cortex. *Journal of Neuroscience*, 27(40):10742–10750, 2007.

[41] Kyle Aitken and Stefan Mihalas. Neural population dynamics of computing with synaptic modulations. *Elife*, 12:e83035, 2023.

[42] Kyle Aitken, Luke Campagnola, Marina E Garrett, Shawn R Olsen, and Stefan Mihalas. Simple synaptic modulations implement diverse novelty computations. *Cell Reports*, 43(5), 2024.

[43] Nicolas Frémaux and Wulfram Gerstner. Neuromodulated spike-timing-dependent plasticity, and theory of three-factor learning rules. *Frontiers in Neural Circuits*, 9:85, 2016.

[44] Yuhan Helena Liu, Stephen Smith, Stefan Mihalas, Eric Shea-Brown, and Uygar Sümbül. Biologically-plausible backpropagation through arbitrary timespans via local neuromodulators. *Advances in Neural Information Processing Systems*, 35:17528–17542, 2022.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We believe that the claims made in the introduction (i.e. that the NM-RNN performs and generalizes better than the LR-RNN) are adequately shown by the results.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We mention limitations of our approach in the Discussion section of the main paper.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: All proofs and derivations are included in full in the Supplementary Material.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We describe the methods by which we generate synthetic data trials in the main text and Supplementary Material. We also include specific training details and hyperparameters in the Supplementary Material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The necessary code required to reproduce the trained models is included in the Supplementary Material. This code will be made available in a public repository upon publication.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: All training/test details are included in the Supplementary Material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: For most of our results, all performance datapoints are presented, eliminating the need for error bars. All other statistical information is included accurately in the text.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Descriptions of the required compute resources used are included in the Supplementary Material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the Code of Ethics and believe our paper conforms to it.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: This paper dealt with modeling synthetic data, with no current real-world use cases. While there are no immediate societal impacts that we anticipate, we believe this kind of work could contribute to better understanding the human brain, leading to improved treatment for disorders.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.