Robust Working Memory through Short-Term Synaptic Plasticity

Leo Kozachkov[1,2,3,*], John Tauber[1,2,4*], Mikael Lundqvist[1,2,5], Scott L. Brincat[1], Jean-Jacques Slotine[2,3,], Earl K. Miller[1,2]

* Co-first authors

1. The Picower Institute for Learning & Memory, Massachusetts Institute of Technology (MIT), Cambridge, MA 02139, USA

2. Department of Brain & Cognitive Sciences, Massachusetts Institute of Technology (MIT), Cambridge, MA 02139, USA

3. Nonlinear Systems Laboratory, Massachusetts Institute of Technology (MIT), Cambridge, MA 02139, USA.

4. Institute for Data, Systems, and Society, Massachusetts Institute of Technology (MIT), Cambridge, MA 02139, USA.

5. Department of Clinical Neuroscience, Karolinska Institute, Stockholm 171 77, Sweden.

Abstract

Working memory has long been thought to arise from sustained spiking/attractor dynamics. However, recent work has suggested that short-term synaptic plasticity (STSP) may help maintain attractor states over gaps in time with little or no spiking. To determine if STSP endows additional functional advantages, we trained artificial recurrent neural networks (RNNs) with and without STSP to perform an object working memory task. We found that RNNs with and without STSP were both able to maintain memories over distractors presented in the middle of the memory delay. However, RNNs with STSP showed activity that was similar to that seen in the cortex of monkeys performing the same task. By contrast, RNNs without STSP showed activity that was less brain-like. Further, RNNs with STSP were more robust to noise and network degradation than RNNs without STSP. These results show that STSP can not only help maintain working memories, it also makes neural networks more robust.

Introduction

Working memory (WM), the holding of information “online” and available for processing, is central to higher cognitive functions[1,2]. Its well-established neural correlate is spiking over a memory delay[3–5]. For many years, this was thought to be the sole mechanism underlying WM maintenance. The idea is that sensory inputs elicit unique patterns of spiking that are sustained via recurrent connections[6], creating attractor states—stable patterns of activity that retain the WM[7]. It seems evident that these attractor dynamics play an important role in WM. Recent

observations, however, have suggested that there may be more going on[8–11]. A few neurons seem to show spiking that looks persistent enough to be an attractor state, but the bulk of neurons show memory delay spiking that is sparse[12–15]. This is especially true when spiking is examined in real time (i.e., on single trials) because averaging across trials can create the appearance of persistence even when the underlying activity is quite sparse. Or it might just be that attractor dynamics are more a phenomenon of populations of neurons rather than single neurons. However, this spareness is also evident in local circuits. In the prefrontal cortex (thought to be central to WM), bursts of spiking (and gamma power associated with it) are coordinated among local neurons [14].

This all begs the question of how WMs are maintained over these gaps in time with little-to-no spiking[11,16]. One possibility was suggested by observations of short-term synaptic plasticity (STSP) in circuits in the prefrontal cortex, transient (< 1 second) changes in synaptic weights induced by spiking[17]. Several groups have suggested updating the attractor dynamics model with this feature[18–20]. The idea is that STSP helps the spiking. Spikes induce a transient "impression" in the synaptic weights that can maintain the network state between spikes [21–23]. Evidence for STSP comes from techniques like patch-clamp recording that are difficult to implement in the working brain, especially in non-human primates (NHPs). Thus, we tested the role of STSP in WM by using computational modeling in conjunction with "ground-truthing" via analysis of spiking recorded from the PFC a NHP performing a WM task. We aimed to determine if network models with STSP can solve the working memory task, whether they have properties similar to those seen in the actual brain, and whether STSP endows functional advantages. The answer to these questions was "yes".

We constructed Recurrent Neural Networks (RNNs) with and without STSP to test how it affects network performance and function. We focused on the key property of robustness[9,24–27]. Working memories must be maintained in the face of distractions like new inputs. Networks need to deal with noise and show graceful degradation (i.e., continue to function when portions of the network are damaged). Our analysis showed that RNNs both with and without STSP were robust against distractors. However, *only the RNNs with STSP were "brain-like"* --their activity more closely resembled activity recorded from the prefrontal cortex of monkeys performing a WM task. RNNs with STSP were also more robust against increasing levels of noise and synaptic ablation. Thus, STSP not only explains how WM can be maintained between spiking, it also carries additional functional advantages.

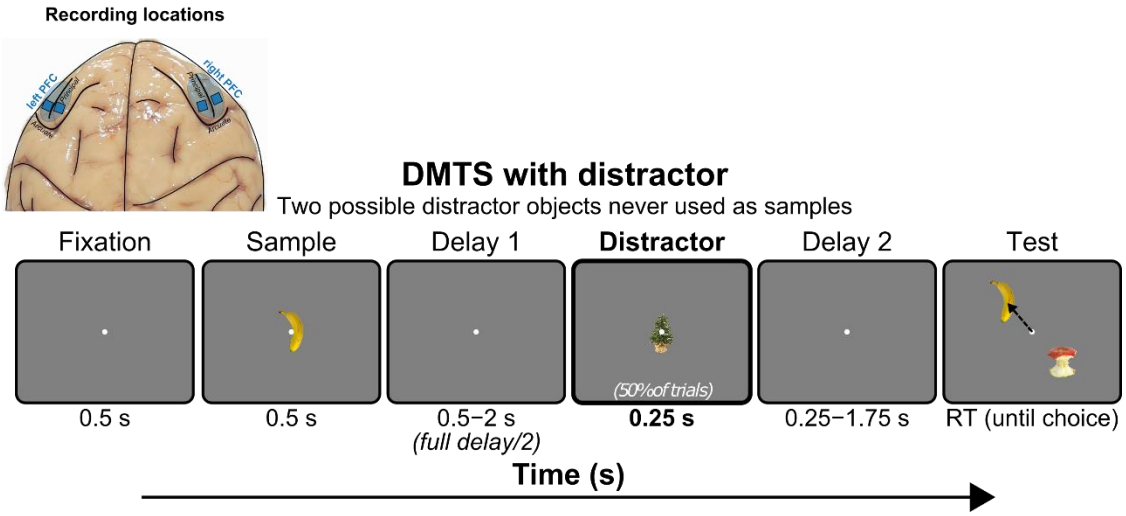

Figure 1: Electrode location and task structure. Utah arrays were implanted bilaterally in dorsolateral PFC (dlPFC) and ventrolateral PFC (vlPFC). Animal performed a distracted delayed match-to-sample task. Each trial began with visual fixation on the middle of the screen for 0.5s. Fixation was maintained throughout the trial until the behavioral response. The delay length was parametrically varied from 1 – 4 s in five logarithmic steps, randomly chosen each trial. At mid-delay a neutral distractor (1 of 2 possible objects never used as samples) was presented randomly on 50% of trials. During the multi-choice test the monkey was allowed to freely saccade between all objects on the screen. The final choice was indicated by fixating on it for at least one second.

Results

A monkey was trained to perform an object delayed-match-to-sample task (Figure 1). They were shown a sample object and had to choose its match after a variable-length memory delay. At mid-delay a distractor object (1 of 2 possible objects never used as samples) was presented (for 0.25s) on 50% randomly chosen trials. We recorded multi-unit activity (MUA) bilaterally in dorsolateral PFC (dlPFC) and ventrolateral PFC (vlPFC) using four 64-electrode Utah arrays for a total of 256 electrodes. The animal learned to do the task consistently at ~99% accuracy for both distractor and non-distractor trials

Sample Information in Population Neural Activity Was Weak Over Longer Delays

First, we examined MUA recorded from the lateral PFC. To quantify the amount of sample object information carried by spiking, we used a linear classifier (see methods for details)[1]. This

showed that spiking carried the most sample object information for about one second after the sample disappeared. For longer delays, there was less sample object information (Figure 2, supp). We corroborated this by measuring the distance between neural population activity for all pairs of sample objects. That gave an average distance between experimental conditions at every timepoint (see methods). This showed the distance between population MUA activity for different samples returned to pre-stimulus levels (Figure 2b). Interestingly, we found that this was *not* simply due to spiking returning to pre-sample values. We determined this by training a classifier to discriminate between pre and post sample spiking activity. We found that this classifier was consistently able to discriminate between pre and post sample spiking activity over the delay (supp Figure 8).

Population Neural Activity Was Robust to Distractors

We found that when the mid-delay distractor was presented, neural trajectories diverged from that of non-distractor trials, as expected (Figure 2a,c, supp). Once the distractor disappeared, the trajectories quickly reconvened, indicating neural stability. This was true for all five delay lengths used (1 – 4 s in five logarithmic steps, supplementary materials). The time course of trajectory reconvening was roughly exponential in form with a time constant of ~200 milliseconds (Figure 2c, supp). We determined this by fitting an exponential function to the recovery curve and measuring the inverse of the fitted decay constant. This is consistent with prior observations that time constants in cortex peak at this value in the prefrontal cortex [28].

This raises a couple of questions. 1. How can PFC networks support WM task performance when, across the population, sample information in spiking can be relatively weak? 2. How do PFC networks achieve the stability to recover from distractors? To answer them, we used RNN modeling and neural network theory. As we will show, the two questions share a common answer: STSP.

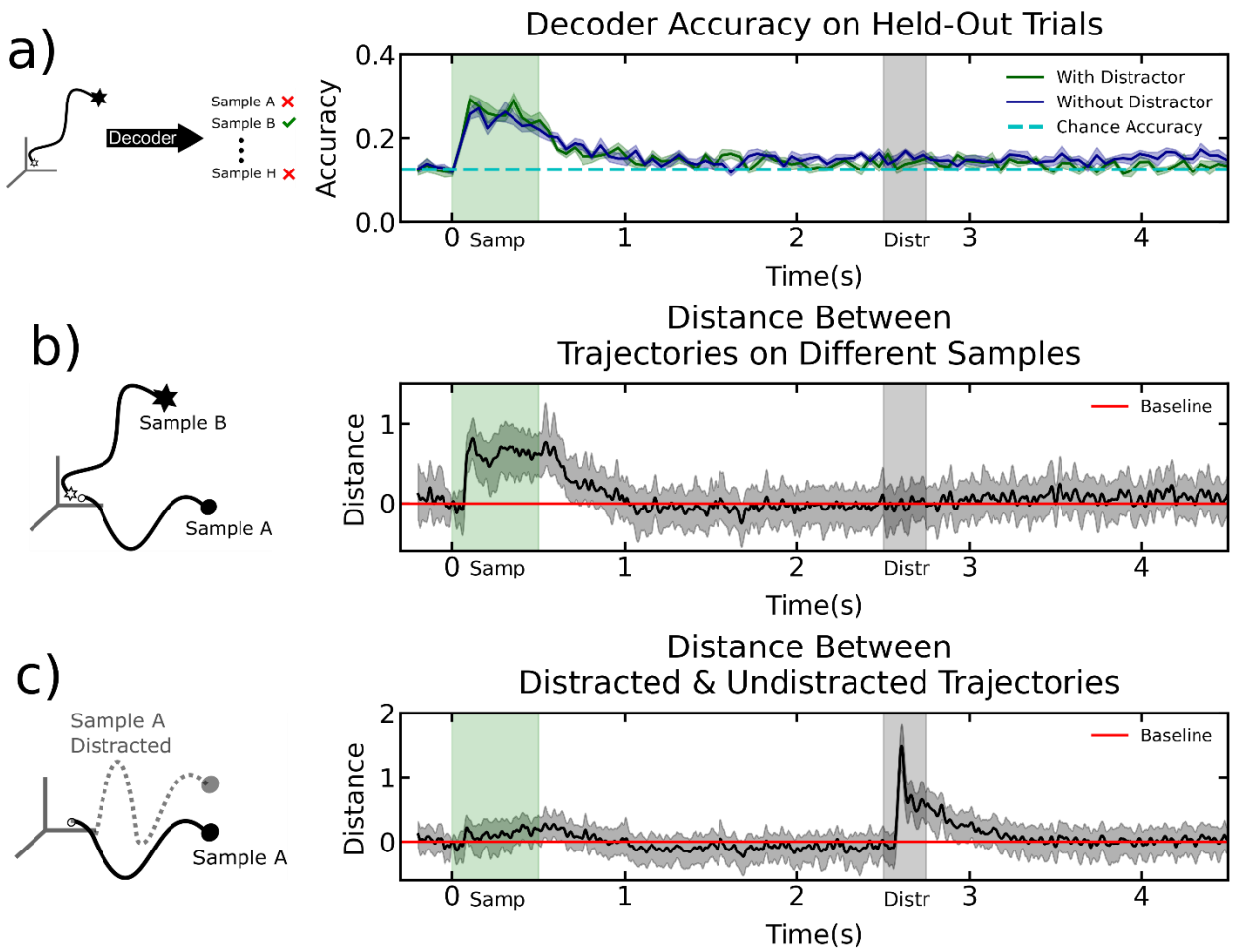

Figure 2: a) Left: training a decoder to predict sample identity given a neural trajectory. Right: The decoder accuracy on held-out trials for distracted and undistracted trials b) Left: comparing trial-averaged trajectories corresponding to different samples. Right: The average pairwise distance in state-space between trajectories elicited by all possible sample images. Normalized by the average pre-stim distance. c) Left: comparing trial-averaged distracted vs. non-distracted trajectories through neural state space. Right: The distance in state space between distracted vs. non-distracted trajectories throughout the trial. Shown are trials with a delay of four seconds.

RNNs With STSP are More Brain-Like

RNNs with and without STSP were able to successfully perform the object delayed match to sample task. However, only the RNNs with STSP did so with activity that was similar to that seen in actual PFC neurophysiological data. Our hypothesis space consisted of four different kinds of RNNs: two with fixed synaptic weights (following training) and two with STSP. The two fixed-weight networks were 'vanilla' recurrent neural networks. All networks had 100 neurons. They differed only in their activation functions. One used a hyperbolic tan (tanh), the other used a rectified linear (ReLU). We will refer to these fixed synapse networks as *FS-tanh* and *FS-relu*

respectively. The synaptic weights for the fixed-synapse networks (without STSP) were only adjusted during training. After training, they are untouched. The other two models use STSP to adjust synaptic weights in an online fashion during each trial. One RNN with STSP was based on a model introduced by Mongillo et al[19]. The model used synaptic calcium dynamics for STSP. It was trained with deep learning using techniques employed by Masse[29]. The Mongillo (*PS-pre*) model adjusts synaptic weights based on presynaptic neural activity. The other RNN with STSP was introduced by Kozachkov et al[30] (*PS-hebb*). It used excitatory anti-Hebbian and inhibitory Hebbian mechanisms to endow network stability. We refer the reader to the Methods section for more details on the models and the training process.

We found that the *FS-tanh* and *FS-relu* fixed-synapse RNNs learned to perform the task by using simple attractors in state space (Figure 3). We measure the distances between state-space trajectories, as we did for the neural data. This revealed different trajectories for different sample objects. Each trajectory settled into a steady state that was unique for each sample, indicating the presence of attractors (Figure 3). Comparison between trials with and without a distractor showed that the distractor temporarily knocked trajectories out of the attractor state. Then trajectories quickly returned to the pre-distractor attractor state. This is how the fixed-synapse RNNs achieve robustness. Once the neural trajectories were 'captured' by the appropriate attractor, they remained fixed there and can return there after being perturbed. In terms of neural activity this means that the spike rates stayed at a particular, elevated firing rate across the delay period corresponding to the sample identity. In other words, these models achieve robust WM through persistent neural activity as a consequence of attractor dynamics, as observed in many prior models.

The models with STSP did not achieve robust WM through persistent neural spiking, but instead relied on the STSP. We again examined trajectories using spikes rates from the STSP models (Figure 4). The neural distance between trajectories for different samples increased during sample presentation but then dropped back down near zero during the delay, especially longer delays. This mirrors the results from actual MUA activity in the PFC (see above). This stands in contrast to the fixed-synapse models (above), in which strong persistent spiking was unlike that observed in the brain. However, unlike the fixed synapse models, in the STSP models we could use synaptic weights to measure trajectories and distance in *synaptic* state-space (i.e., decode the sample identity). This is analogous to using spike rates but instead we measure synaptic weights over time (as was done in Masse[29]). This revealed that trajectories for different

samples in synaptic state-space remained elevated across the delay period (Figure 4), indicatingsample information was being maintained by STSP even when spiking levels were low.

As we found in the experimentally recorded MUA, in the STSP models the spike rate trajectories between distractor and non-distractor trials increased during distractor presentation but then quickly decreased back to pre-distraction levels (Figure 4). By contrast, the distractor had a longer lasting effect in synaptic weight space for the STSP models. However, and notably, this did not affect the ability of the STSP networks to perform the task. The STSP networks performed at a high level (above 90% accuracy) on both distractor and non-distractor trials.

As in analysis of the experimental data, we also used a decoder to read out sample object information over time for the fixed-synapse and STSP models. For the fixed-synapse RNNs we trained the decoder on trajectories from spike rates. For the STSP RNNs, we trained two decoders: one on the spike-rate trajectories and one on the synaptic weight trajectories. In both classes of models, sample presentation elicited a large increase in decoding accuracy (Fig 5). Spike rate decoding of sample information in the fixed-synapse models remained high throughout the memory delay. This may seem advantageous, but it is in sharp contrast to the actual PFC MUA data. The PFC MUA showed a sharp drop in sample object decoding accuracy once the sample disappeared and especially for delays longer than one second. This was mirrored in the STSP models (Fig 5). However, sample decoding accuracy in synaptic weight space for the STSP models remained high over the entire delay, including longer delays (Figure 5).

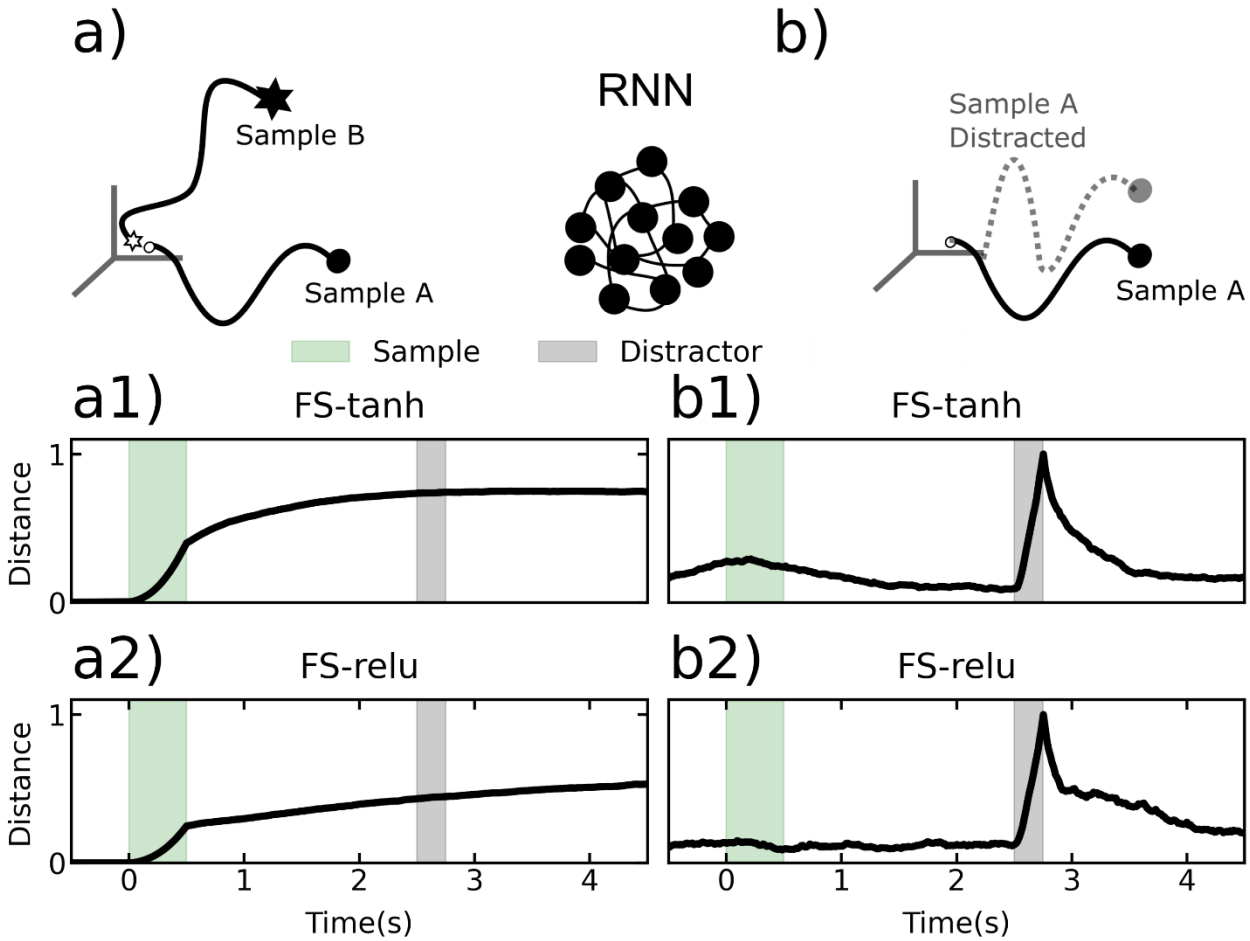

Figure 3: a) Cartoon of trial-averaged RNN trajectories corresponding to two different sample conditions for the fixed-synapse RNNs. a1) Average pairwise distance between trajectories on different sample conditions, for the fixed synapse network with tanh activation (FS-tanh). a2) The same plot as in a1, but for FS-relu. b1) The average distance between distracted and undistracted trajectories. Average taken over all sample conditions. Results shown for FS-tanh. b2) Same results as in b1, but for FS-relu.

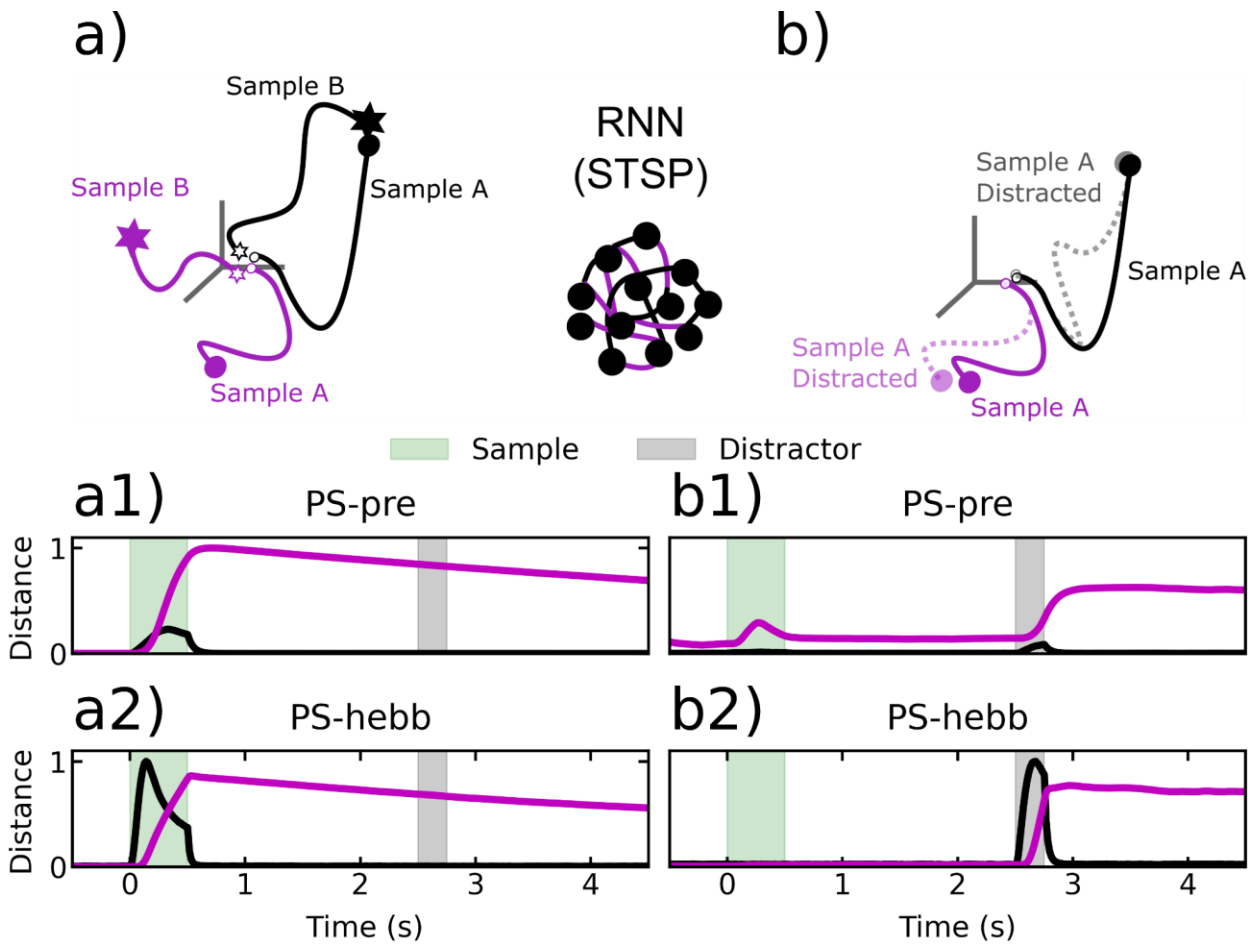

Figure 4: a) Cartoon of trial-averaged neural and synaptic RNN trajectories corresponding to two different sample conditions for the plastic-synapse RNNs. Black lines correspond to neural trajectories, purple lines correspond to synaptic trajectories. a1) Average pairwise distance between neural and synaptic trajectories on different sample conditions, for PS-pre. a2) The same plot as in a1, but for PS-hebb. b1) The average distance between distracted and undistracted trajectories. Average taken over all sample conditions. Results shown for PS-pre. b2) Same results as in b1, but for PS-hebb.

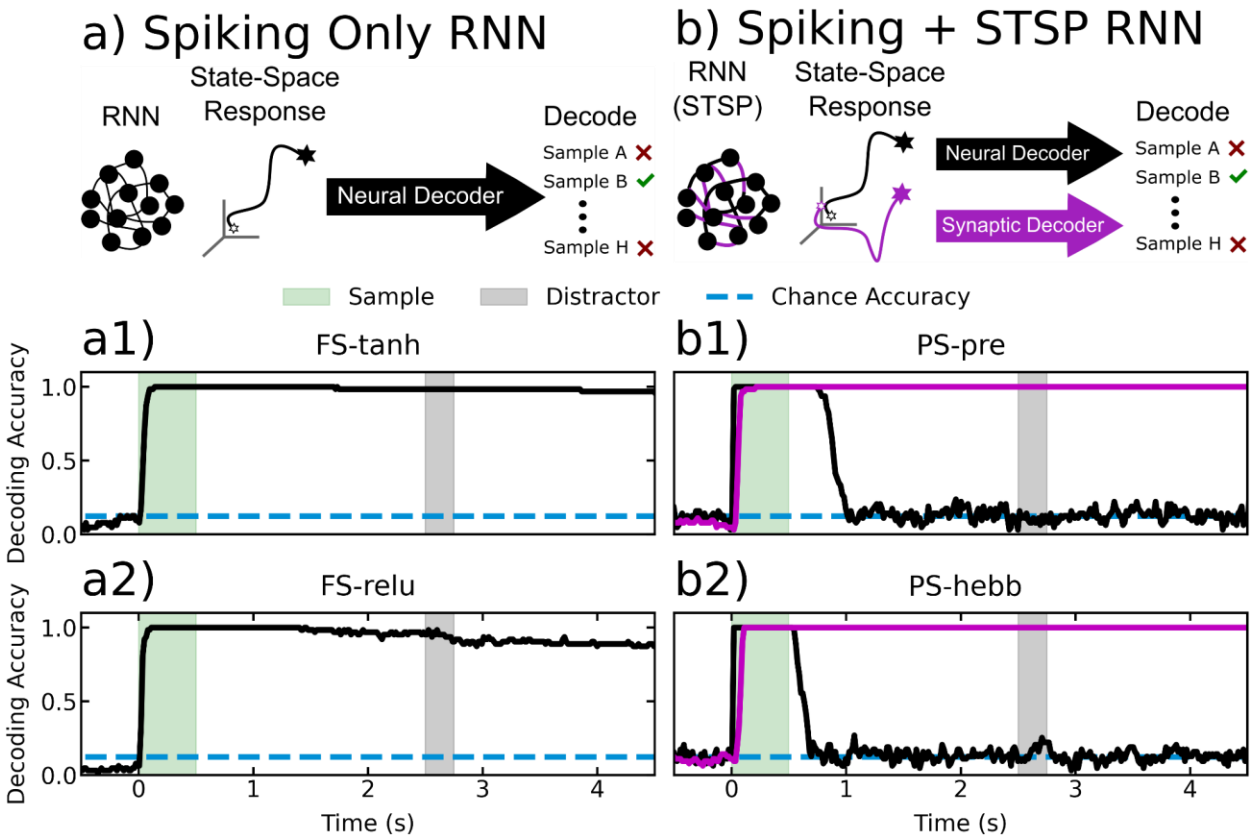

Figure 5: Decoding accuracy for RNNs. a) A decoder is trained to predict the sample label from RNN neural trajectories, for the fixed-synapse RNNs. a1) The accuracy of the trained decoder as a function of time from sample onset for FS-tanh. a2) The same plot as a1, for FS-relu. b) Two decoders are trained on the neural and synaptic trajectories separately. Black lines indicate neural decoding, purple indicate synaptic decoding. b1) Neural and synaptic decoding accuracy as a function of time for PS-pre. b2) Same plot as in b1 but for PS-hebb.

STSP Increases Structural Robustness

The two models with STSP were *structurally* more robust than the models with fixed synaptic weights. To measure robustness, we looked at how the performance accuracy of the trained network varied as we randomly ablated a varying fraction of synaptic weights. This showed that the STSP network's performance of the task remained high even when as many as half of the synapses were ablated (Figure 6). By contrast, the fixed-synapse models were highly sensitive to synaptic ablation. Their performance severely degraded even if only 10-20% of the synapses were ablated.

All models were qualitatively similar in their sensitivity to noise. As we increased the amount of process noise (see methods for definition), the STSP model *PS-hebb* was revealed to be the most robust (Figure 6).

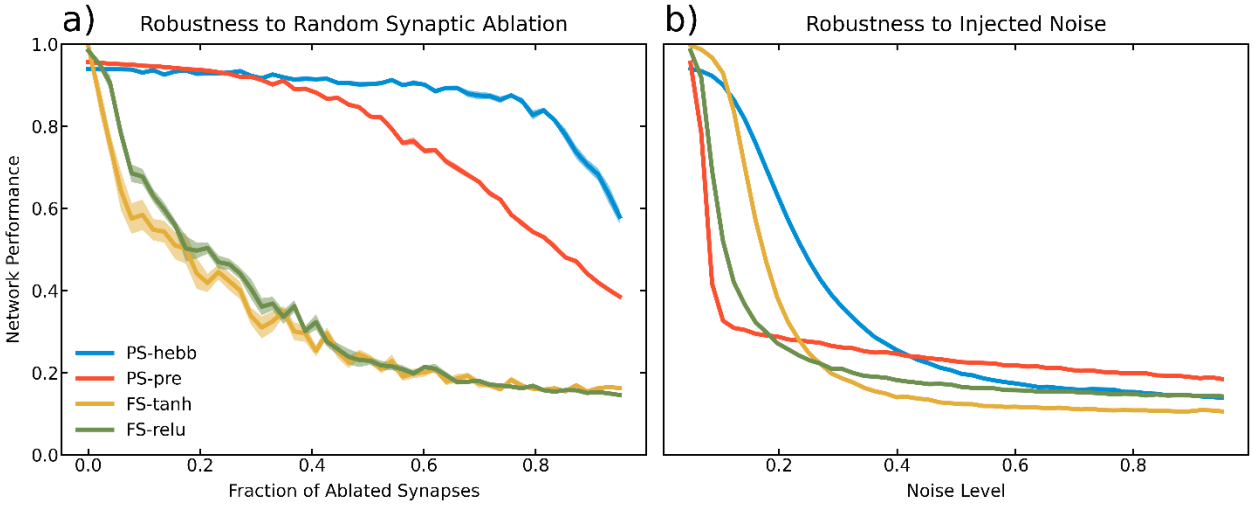

Figure 6: a) RNN test set performance (fraction of correct trial over test set) as function of the fraction of randomly ablated synapses. Shaded regions denote standard error computed over 10 repetitions. b) Same as in (a), but as a function of increasing process noise.

Discussion

We found that RNNs with and without short-term synaptic plasticity (STSP) both show robustness against distractors. However, STSP network models showed graceful degradation to synaptic loss. They were still able to function even after half of the synapses were lost. By contrast, fixed-weight attractor dynamic models did not. They showed a severe decrease in function with as little as 10-20% synaptic loss. STSP models also showed more robust performance in response to noise. In the STSP model, spike rates decreased during the memory delay (especially longer delays). This reduced the ability to decode WM from spiking but, importantly, the WMs could be decoded from the synaptic weights. This contrasts with RNN models without STSP where spiking activity remained high over the entire memory delay. That, however, did not match observations of actual PFC spiking recorded from the prefrontal cortex during an object delayed match-to-sample-task. PFC spiking showed a decrease during the memory delay that mirrored that of the STSP models. In sum, STSP can not only maintain information over gaps of no spiking, it also adds functional advantages. And, notably, adding STSP to RNNs makes them function more like actual brain networks.

Some forms of STSP worked better than others. We found that purely Hebbian STSP models were difficult to train, in agreement with previous studies[31]. By contrast, an anti-Hebbian STSP model (which reduced synaptic weights when spiking was too correlated) was successful and did not require any external weight clipping. We and others have found that anti-Hebbian STSP plays a crucial role in network stability and function[30,32]. And like others, we found that the synaptic weights in the *PS-pre* model had to be limited to a certain range to maintain stability and trainability[19,29]. This assumption is biologically plausible. Biological synapses are limited in the amount of resources that can be recruited by a presynaptic spike[19,33].

The STSP models are in line with a variety of studies suggesting that cognition is more complex than steady attractor states. For example, sustained attention was long-thought to depend on attractor-like steady-state spiking. However, like studies of WM, this may have been an artifact of averaging spiking across trials[11,14]. Examination of sustained attention in real time (on single trials) has shown that, behaviorally, attention waxes and wanes rhythmically at theta[34]. There is a corresponding waxing and waning of spiking synchronized to LFP theta oscillations. Likewise, neural correlates of WM show sparse bursts of spiking linked to oscillatory dynamics when examined in real time[14,15]. Further, the degree of memory delay spiking in the PFC is highly dynamic and task-dependent. For example, Masse found that models with STSP use higher spiking levels when WMs are being manipulated. This is consistent with Lundqvist et al[14], who found spiking increases when WMs are about to be used (e.g., near the end of a memory delay). This increased spiking is an important feature of the STSP models because the spiking is needed to read out information from the pattern of synaptic weights. A recent paper analyzed spiking from dlPFC during a fixed-delay spatial delayed response task and found significant persistent delay activity[10]. Our WM task had variable delays. We did not observe persistent neural delay activity. It could be that the PFC does a kind of cost/benefit analysis to determine the degree of persistent delay activity to use, depending on the task structure. A recent computational study by Orhan and Ma found that fixed-synapse RNNs trained on a short-term memory task with variable delays exhibited more persistent activity than the same RNNs trained with fixed delay[31]. It would be interesting to see if/how this result changes in RNNs with STSP.

It is also worth noting that studies that report more memory-delay spiking tend to be studies that used spatial delayed response tasks. It has been argued that spatial delayed response is more of a motor inhibition task than a WM task[15]. The animal knows the behavioral response to be

executed and is inhibiting it while waiting for a “go” signal to respond. In WM tasks like object delayed match-to-sample (i.e., the task used in this study), by contrast, the behavioral response is not known until after the memory delay. They may be more “pure” WM tasks in that they involve holding information for further computation, not inhibiting a behavioral response. Tasks like delayed match-to-sample tend to produce lower levels of memory delay spiking[14,15,29,35] and thus require more than attractor states alone.

The robustness and stability added by STSP is critical not just to WM but top-down control in general. More broadly, if one views the PFC as the brain’s control hub[2], then robustness is a bare necessity. A control system which is overly sensitive to perturbations would quickly propagate these disturbances to the rest of the brain, leading to errant behavior. Moreover, robustness and stability are intimately tied to modularity[36–38]. Stable networks can be linked with other stable networks in a way that preserves stability. Without that property, the whole network must be retrained when new modules are added to subserve complex or composite behavior. Our results suggest that STSP has a dual role. It can maintain information while simultaneously ensuring robustness in an energy-efficient manner[9]. We speculate that stability could be a key component to understanding the way specialized modules in the brain dynamically cooperate to form cohesive perceptions, plans, and actions.

Methods

Subject and Task

The non-human primate subject used in our experiments was a male rhesus macaque monkey (*Macaca mulatta*), aged 17. All procedures followed the guidelines of the Massachusetts Institute of Technology Committee on Animal Care and the National Institutes of Health.

As described in the main text, Utah arrays were implanted bilaterally in dorsolateral PFC (dlPFC) and ventrolateral PFC (vlPFC). Animal performed a distracted delayed match-to-sample task. Each trial began with visual fixation on the middle of the screen for 0.5s. Fixation was maintained throughout the trial until the behavioral response. The delay length was parametrically varied from 1 – 4 s in five logarithmic steps, randomly chosen each trial. At mid-delay a neutral distractor (1 of 2 possible objects never used as samples) was presented randomly on 50% of trials. During the multi-choice test the monkey was allowed to freely saccade between all objects on the screen. The final choice was indicated by fixating on it for at least one second.

Data Analysis Methods

Distances Between Neural Trajectories

To quantify the difference in population activity between different conditions, we computed the distance between neural trajectories. We define a neural trajectory as a vector of firing rates for N recorded neurons over time.

For the distractor vs non-distractor comparison, denote $x_t^{s,d,l} \in [0,\infty)^N$ the neural trajectory at time t on trial $l \in \{1,\dots,L\}$ for sample object $s \in \{1,\dots,S\}$ and where $d \in \{0,1\}$ denotes the presence or absence of a distractor in the delay phase. We first computed trial averages as

$$\bar{x}_t^{s,d} = \sum_{l=1}^{L} x_t^{s,d,l}$$

Then for each s we computed the distance between distracted and non-distracted trajectories, using the Euclidean norm, and took the average over sample objects:

$$d_t^{\text{dist}} = \sum_{s=1}^{S} \left\| \bar{x}_t^{s,1} - \bar{x}_t^{s,0} \right\|^2$$

For the comparison between trajectories of different sample objects, we used only non-distractor trials ($d = 0$). We first averaged over trials as above for each sample object to get

$$\bar{x}_t^{s} = \sum_{l=1}^{L} x_t^{s,0,l}$$

and calculated the mean distance between each pair of stimuli as:

$$d_t^{\text{stim}} = \sum_{1 \le i < j \le S} \left\| \bar{x}_t^{i} - \bar{x}_t^{j} \right\|^2$$

Confidence Intervals for Trajectory Distances

As our final measurement we calculated the mean over six experimental sessions for both d_t^{dist} and d_t^{stim}. To quantify uncertainty that captures both the session and trial variability, we used a non-parametric multi-level bootstrap [39]. Briefly, for $b \in 1,\dots,B$ with $B = 1000$, we first sampled with replacement over sessions (that is, we randomly selected a session with replacement E times, where E is the total number sessions in the dataset - in our case $E = 6$), and then for each resampled session, we resampled trials with replacement in the trial-average steps described above, yielding the bootstrap samples $d_t^{\text{dist},b}$ and $d_t^{\text{stim},b}$. We took the 2.5^{th} and 97.5^{th} percentiles across b as our lower and upper values for the 95% confidence interval.

Sample Object Decoding

For sample object decoding experiments, we first smoothed trials (as above, with 10ms Gaussian kernel), and then took average rates per neuron in 50ms time bins and attempted to decode the

sample object from population activity at each time point. For our classifier, we used a linear Support Vector Machine and standard regularization (C=1 in SciKit Learn). For each session we used a 10-fold cross-validation to calculate decoding accuracy. (That is, we held out 10% of trials as a test set and trained on the remaining 90% of the data. We did this for 10 non-overlapping test sets, and then took the average test accuracy). We performed a decoding analysis on each session separately, and then took the average and standard error across sessions for our result.

Change from Baseline Decoding

We smoothed trials and took average rates as in the sample decoding above. We performed 10-fold cross-validation as above for calculating test accuracy, and calculated the average performance over samples for each session. Plots show the average and standard error across sessions.

Artificial Neural Network Modelling

Fixed-Synapse (FS)

For the fixed-synapse ANNs we used a 'vanilla' recurrent neural network. Each neuron had an activation x_i which we collected into a vector $x \in R^n$ such that $(x)_i = x_i$. This vector was the *state* of the neural network at a given time *t*. The state evolves in time according to the following differential equation:

$$\tau\dot{x} = -x + \phi(Wx + I + b + Noise)$$

Here τ is the time-constant of the network, which we set to 100 milliseconds. The function $\phi(\cdot)$ is a nonlinearity (e.g tanh, softplus, etc). The matrix W is the recurrent weight matrix of the network, and determines how the different neurons interact with one another. The vector $I \in R^n$ is the exogenous input into the recurrent neural network, and can vary with *t*. For our purposes I will represent sensory information related to the task at hand. The vector $b \in R^n$ is a fixed bias term. The noise will be defined shortly. To integrate the neural dynamics, we use the standard Euler approximation scheme with fixed step-size of dt = 15 milliseconds. Defining the ratio of dt to τ as $\alpha \equiv \frac{dt}{\tau}$, the approximated dynamics are:

$$x_{t+1} = (1 - \alpha)x_t + \alpha\phi(Wx_t + I_t + b + Noise)$$

Here $Noise = \sqrt{(2\alpha^{-1}\sigma_{rec}^2)}N(0,1)$ where $\sigma_{rec} = 0.05$ denotes the *process noise* and $N(0,1)$ denotes a draw from the standard normal distribution. For the FS-tanh networks, we use $\phi(\cdot) = \tanh(\cdot)$. For the FS-ReLU networks, we use $\phi(\cdot) = \mathrm{ReLU}(\cdot) = \max(0,\cdot)$. Finally, we used an affine readout for the network to produce the desired output for the task:

$$y = W_{out}x + c$$

With the exception of W, all the parameters of the fixed-synapse networks were initialized by drawing each element randomly from a gaussian distribution with mean zero and standard deviation $\frac{1}{\sqrt{n}}$. For the recurrent weight matrix W the elements were drawn from a gaussian distribution with mean zero, but standard deviation $\frac{0.9}{\sqrt{n}}$. We found that this initialization achieved faster training. At the beginning of each trial, the network state was initialized at the origin (i.e $x = 0$).

Plastic Synapse

PS-Pre

This ANN was an implementation of the model originally introduced by Mongillo et al [19] and then subsequently trained using deep learning techniques by Masse et al [29]. Many of the details are similar to or exactly the same as in Masse et al. We go through them here for completeness. As in the fixed-synapse model, we used Euler integration to approximate the dynamics:

$$x_{t+1} = (1 - \alpha)x_t + \alpha\phi(Wx_t u_t a_t + I_t + b)$$

Aside from the addition of two new variables $u_t$and a_t (which we'll define shortly) this is the same model as the fixed-synapse model. This network consisted of 80% excitatory neurons and 20% inhibitory neurons. To keep these populations separated throughout training, we decomposed Was $W = W_{\boldsymbol{rec}}^{+}D$ where $W_{\boldsymbol{rec}}^{+}$ is a matrix with non-negative entries and D is a diagonal matrix whose ii^{th} entry is either 1 or -1, depending on whether neuron i is an excitatory or inhibitory neuron. To each synapse we associated a value a, for the fraction of available neurotransmitter and u, the amount of utilized neurotransmitter. These two variables evolved according to:

$$\dot{a} = \frac{1 - a}{\tau_a} - u(t)a(t)x(t)dt$$

$$\dot{u} = \frac{U - u(t)}{\tau_u} + U\big(1 - u(t)\big)x(t)dt$$

here $x(t)$ is the presynaptic activity at time t, τ_a is the time constant of available neurotransmitter recovery, τ_u is the time constant of neurotransmitter utilization recovery. Half the synapses in the network were facilitating, the other half was depressing. For the facilitating synapses, $\tau_a = 200\ ms$, $\tau_u = 1500\ ms$ and $U = 0.15$. For the depressing synapses, $\tau_a = 1500\ ms$, $\tau_u = 200\ ms$ and $U = 0.45$. Aside from $W_{\boldsymbol{rec}}^{+}$, the parameter initialization for PS-pre was the same as the fixed synapse networks. For W, we found that the following initialization worked best in terms of achieving the fastest training time:

1) Draw each element of $W_{\boldsymbol{rec}}^{+}$ from the log-normal distribution, with the underlying normal distribution having mean 0 and standard deviation $\frac{0.9}{\sqrt{n}}$.
2) Compute the largest singular value of $W_{\boldsymbol{rec}}^{+}$ (call it γ) and divide every element of $W_{\boldsymbol{rec}}^{+}$ by 10γ.

To ensure that W_{rec}^{+} stayed non-negative throughout training, we always ran each element through a rectified linear function at the beginning of the trial. For each trial, the neural state vector x was initialized at the origin. The facilitation variables a were all initialized at $a = 1$. The utilization variables were all initialized at $u = 0$. Throughout the trial, both facilitation and utilization were clamped between 0 and 1. The readout from the network was affine, as in the fixed-synapse networks.

PS-Hebb

This excitatory anti-Hebbian / inhibitory Hebbian ANN was originally introduced in Kozachkov et al [30]. As before, we use a Euler integration scheme to approximate the dynamics:

$$x_{t+1} = (1-\alpha)x_t + \alpha\phi(W_t\, x_t + I_t + b + Noise)$$
$$W_{t+1} = (1-\alpha\gamma\,)W_t\, - \alpha K \circ \mathrm{xx}^{\mathrm{T}}$$

Here $\phi(\cdot) = \cdot$. That is, ϕ is just the identity function. The parameter γ controls the degree of weight decay, and is set to $1/200ms^{-1}$. The matrix K is positive and positive-definite, and the symbol $\circ$ denotes element-wise matrix multiplication (i.e Hadamard product). During training we ensure positiveness and positive-definiteness of K by parameterizing it as:

$$K = B^T B + 10^{-2}O + 10^{-2}I$$

Where $B_{ij} = C_{ij}^2$, $O_{ij} = 1$ I is the identity matrix. Since $B^T B$ is positive semi-definite and non-negative, the matrix O ensures that K is strictly positive, while the identity matrix ensure that K is strictly positive-definite. The training is done on the matrix C. The output from the network is affine, as in all the other networks. All the parameters are initialized the same way as the fixed-synapse networks, except for C. The elements of this matrix are drawn from a uniform distribution between -0.5 and 0.5. At the beginning of each trial, the neural state vector x and the weights W are initialized at the origin.

Task Structure and Training Details for Artificial Neural Networks

The basic task structure for the ANNs was the same as for the non-human primate. Each stimuli was encoded in a one-hot vector. There was a total of 10 images (8 sample images, 2 distractor). The number of input neurons going into the ANNs was 11 = 10 + 1: 10 input neurons for the stimuli and 1 for a fixation signal. These input stimuli were collected into a vector m_t, which changed with time throughout the trial. This vector was fed into the ANNs via a trainable input weight matrix $I_t = W_{in}m_t$. There were 11 = 8 + 2 + 1 output neurons leaving the ANNs, 8 for each sample image, 2 for the distractors and one as a fixation neuron. These neurons were collected into the vector $y_{desired}$, which defined the desired output of the ANNs.
At $t = 0$ the ANN has to fixate by having its fixation output neuron hold a value of ‘1’. The fixation is left on for 1000ms. At the end of the fixation period a randomly chosen sample image is presented for 500ms. Following the sample presentation period, the delay period begins. The delay length is randomly chosen from 1.0 s, 1.41s, 2.0 s, 2.83 s and 4s.
On 50% of the trials a distractor is presented in the middle of the delay for 250ms. Following the end of the delay period, the test period begins. During the test period the original sample image

is presented as well as an off-target image for 500ms. At this point the RNN must do two things: stop fixating (i.e make the output fixation neuron output a value of '0') and indicate the correct choice by making the output neuron corresponding to the sample image hold a value of '1'.
We use a mean-squared loss to quantify network performance. For each trial we take the network output for a period of 500ms after the test period ends and compute the loss:

$$\mathcal{L} = \frac{1}{n_{out}T}\sum_{t}^{t+T}\left|\left|y - y_{desired}\right|\right|^2$$

We compute this loss for each trial in the batch, and then average over the batch to get a total loss per batch. Our training and test sets consisted of 2^{14} trials each. We used a batch size of 256 to train and test. To train the networks we use the Adam optimizer with Pytorch standard parameters, a weight-decay value of 10^{-4} for all models. For the FS models we use a learning rate of 10^{-3} . For PS-pre we use a learning rate of 0.02. For PS-hebb we use a learning rate of 0.01.
To test network performance, we compute the 'decision' of the network as the index of the output neuron which had the highest value. We compute the average accuracy of the network by the following procedure:

1) In the 500ms following the end of the test period, we compare the network output to the desired output for each time-point. We add up the number of times the network makes the right decision in this period, and then divide over the number of time-points to get a single-trial accuracy.
2) We compute this single-trial accuracy for all trials in the test set, and report the final network accuracy on the task as the average single-trial accuracy, taken over the whole test set.

References

1. Baddeley, A. Working memory. *Science* **255**, 556–559 (1992).
2. Miller, E. K. & Cohen, J. D. An integrative theory of prefrontal cortex function. *Annu. Rev. Neurosci.* **24**, 167–202 (2001).
3. Funahashi, S., Bruce, C. J. & Goldman-Rakic, P. S. Mnemonic coding of visual space in the monkey's dorsolateral prefrontal cortex. *J. Neurophysiol.* **61**, 331–349 (1989).
4. Goldman-Rakic, P. S. Cellular basis of working memory. *Neuron* **14**, 477–485 (1995).
5. Fuster, J. M. & Alexander, G. E. Neuron activity related to short-term memory. *Science* **173**, 652–654 (1971).

6. Amit, D. J. & Brunel, N. Model of global spontaneous activity and local structured activity during delay periods in the cerebral cortex. *Cereb. Cortex N. Y. N 1991* **7**, 237–252 (1997).
7. Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci. U. S. A.* **79**, 2554–2558 (1982).
8. Kamiński, J. & Rutishauser, U. Between persistently active and activity-silent frameworks: novel vistas on the cellular basis of working memory. *Ann. N. Y. Acad. Sci.* **1464**, 64–75 (2020).
9. Stokes, M. G. 'Activity-silent' working memory in prefrontal cortex: a dynamic coding framework. *Trends Cogn. Sci.* **19**, 394–405 (2015).
10. Barbosa, J. *et al.* Interplay between persistent activity and activity-silent dynamics in the prefrontal cortex underlies serial biases in working memory. *Nat. Neurosci.* **23**, 1016–1024 (2020).
11. Lundqvist, M., Herman, P. & Miller, E. K. Working Memory: Delay Activity, Yes! Persistent Activity? Maybe Not. *J. Neurosci.* **38**, 7013–7019 (2018).
12. Pasternak, T. & Greenlee, M. W. Working memory in primate sensory systems. *Nat. Rev. Neurosci.* **6**, 97–107 (2005).
13. Cromer, J. A., Roy, J. E. & Miller, E. K. Representation of Multiple, Independent Categories in the Primate Prefrontal Cortex. *Neuron* **66**, 796–807 (2010).
14. Lundqvist, M. *et al.* Gamma and Beta Bursts Underlie Working Memory. *Neuron* **90**, 152–164 (2016).
15. Miller, E. K., Lundqvist, M. & Bastos, A. M. Working Memory 2.0. *Neuron* **100**, 463–475 (2018).
16. Constantinidis, C. *et al.* Persistent Spiking Activity Underlies Working Memory. *J. Neurosci.* **38**, 7020–7028 (2018).
17. Wong, K.-F. & Wang, X.-J. A Recurrent Network Mechanism of Time Integration in Perceptual Decisions. *J. Neurosci.* **26**, 1314–1328 (2006).
18. Lundqvist, M., Herman, P. & Lansner, A. Theta and Gamma Power Increases and Alpha/Beta Power Decreases with Memory Load in an Attractor Network Model. *J. Cogn. Neurosci.* **23**, 3008–3020 (2011).

19. Mongillo, G., Barak, O. & Tsodyks, M. Synaptic Theory of Working Memory. *Science* **319**, 1543–1546 (2008).
20. Sandberg, A., Tegnér, J. & Lansner, A. A working memory model based on fast Hebbian learning. *Netw. Bristol Engl.* **14**, 789–802 (2003).
21. Knoblauch, A., Palm, G. & Sommer, F. T. Memory Capacities for Synaptic and Structural Plasticity. *Neural Comput.* **22**, 289–341 (2010).
22. Seeholzer, A., Deger, M. & Gerstner, W. Stability of working memory in continuous attractor networks under the control of short-term plasticity. *PLOS Comput. Biol.* **15**, e1006928 (2019).
23. Taher, H., Torcini, A. & Olmi, S. Exact neural mass model for synaptic-based working memory. *PLOS Comput. Biol.* **16**, e1008533 (2020).
24. Bouchacourt, F. & Buschman, T. J. A Flexible Model of Working Memory. *Neuron* **103**, 147-160.e8 (2019).
25. Durstewitz, D., Seamans, J. K. & Sejnowski, T. J. Neurocomputational models of working memory. *Nat. Neurosci.* **3**, 1184–1191 (2000).
26. Jacob, S. N. & Nieder, A. Complementary Roles for Primate Frontal and Parietal Cortex in Guarding Working Memory from Distractor Stimuli. *Neuron* **83**, 226–237 (2014).
27. Barak, O. & Tsodyks, M. Working models of working memory. *Curr. Opin. Neurobiol.* **25**, 20–24 (2014).
28. Murray, J. D. *et al.* A hierarchy of intrinsic timescales across primate cortex. *Nat. Neurosci.* **17**, 1661–1663 (2014).
29. Masse, N. Y., Yang, G. R., Song, H. F., Wang, X.-J. & Freedman, D. J. Circuit mechanisms for the maintenance and manipulation of information in working memory. *Nat. Neurosci.* **22**, 1159–1167 (2019).
30. Kozachkov, L., Lundqvist, M., Slotine, J.-J. & Miller, E. K. Achieving stable dynamics in neural circuits. *PLOS Comput. Biol.* **16**, e1007659 (2020).

31. Orhan, A. E. & Ma, W. J. A diverse range of factors affect the nature of neural representations underlying short-term memory. *Nat. Neurosci.* **22**, 275–283 (2019).
32. Tyulmankov, D., Yang, G. R. & Abbott, L. F. Meta-learning synaptic plasticity and memory addressing for continual familiarity detection. *Neuron* (2021) doi:10.1016/j.neuron.2021.11.009.
33. Tsodyks, M. V. & Markram, H. The neural code between neocortical pyramidal neurons depends on neurotransmitter release probability. *Proc. Natl. Acad. Sci.* **94**, 719–723 (1997).
34. Helfrich, R. F. *et al.* Neural mechanisms of sustained attention are rhythmic. *Neuron* **99**, 854–865 (2018).
35. Miller, E. K., Erickson, C. A. & Desimone, R. Neural Mechanisms of Visual Working Memory in Prefrontal Cortex of the Macaque. *J. Neurosci.* **16**, 5154–5167 (1996).
36. Slotine, J.-J. E. & Lohmiller, W. Modularity, evolution, and the binding problem: a view from stability theory. *Neural Netw.* **14**, 137–145 (2001).
37. Slotine, J. E. Modular Stability Tools for Distributed Computation and Control. (2002).
38. Kozachkov, L., Ennis, M. & Slotine, J.-J. Recursive Construction of Stable Assemblies of Recurrent Neural Networks. *ArXiv210608928 Cs Math Q-Bio* (2021).
39. Saravanan, V., Berman, G. J. & Sober, S. J. Application of the hierarchical bootstrap to multi-level data in neuroscience. *Neurons Behav. Data Anal. Theory* **3**, https://nbdt.scholasticahq.com/article/13927-application-of-the-hierarchical-bootstrap-to-multi-level-data-in-neuroscience (2020).