A Unified Dynamic and Computational Mechanism for Persistent and Transient Neural Activity Patterns During Delayed-Response Tasks

Zeyuan Ye[1,2,3,4], Liang Tian[1,5*], Changsong Zhou[1,2,5*]

[1]*Department of Physics, Hong Kong Baptist University, Kowloon Tong, Hong Kong, China*

[2]*Centre for Nonlinear Studies and Beijing-Hong Kong-Singapore Joint Centre for Nonlinear and Complex Systems (Hong Kong), Hong Kong Baptist University, Kowloon Tong, Hong Kong, China*

[3]*Institute of Interdisciplinary Studies, Hong Kong Baptist University, Kowloon Tong, Hong Kong, China*

[4]*Department of Physics, Washington University in St. Louis, St. Louis, Missouri, USA*

[5]*Institute of Computational and Theoretical Studies, Hong Kong Baptist University, Kowloon Tong, Hong Kong, China*

Correspondence and requests for materials should be addressed to L.T. or C.Z. (email: liangtian@hkbu.edu.hk; cszhou@hkbu.edu.hk)

Abstract

Neural activity during short-term memory can be either persistent or transient. However, there is an ongoing debate about which activity pattern presents a more accurate neural basis underlying short-term memory. Here, we addressed this problem by training artificial recurrent neural networks (RNNs) with delayed-response tasks. We found biological features emerged from the trained RNNs. Reverse-engineering showed that persistent and transient neural activity patterns can be unified by a neural state moving on a low-dimensional heteroclinic manifold, driven by a velocity field which is computationally Bayes-optimal to minimize the memory error. Our results shed new light on the neural mechanism and computational principles of short-term memory by unifying two seemly contradictory experimental phenomena into a single framework and suggest a new way to interpret the experimental data using computationally important transient speed as a continuous measure to characterize the neural activities during the delay.

Introduction

Short-term memory is the ability to maintain information within the brain for a short period without external stimuli. It is important for many cognitive functions, such as prediction[1], decision making[2], cognitive control[3,4], and integration[5,6]. The key to understanding short-term memory relies on neural activity during the delay epoch, which is between the stimulus presentation and the behavioural response. Previous studies have observed two distinct patterns of neural activity during the delay epoch: (1) persistent neural activity, in which the neural firing rate is approximately a constant above the baseline[7–15] and (2) transient neural activity, in which the neural firing rates dynamically change[16–24].

Two distinct categories of dynamic models have been proposed to explain the two experimental observations respectively. (1) For persistent activity, a notable model is the discretised attractor model[25–27]. Discretised attractors are different neural populational activity patterns that encode different discretized information values. During the delay, the neural population activity converges to one of the attractors very quickly, therefore generating persistent activity. If the external information is continuous (e.g., directions), the information is stored in continuous attractors[28]. Neural population activity can only diffuse along the continuous attractors slowly[29], which also leads to persistent neural activity on average of repeating trials. However, the continuous attractor system requires sophisticated symmetrical neural connections that is not biologically plausible[30], thus an approximately discretised model has been proposed. A finite number of shallow attractors are used to represent the truncated continuous information values[30–32]. In this case, the neural state is bounded around the attractors; thus, neural activity is also persistent. (2) Models of transient neural activity, in contrast, less directly rely on attractors[19,33–36]. For example, the decaying firing rates of some neurons can be compensated by the increasing firing rates of other neurons, hence a linear combination of all neuronal activities remains invariant to maintain the information[34]. Alternatively, each single discretized information can be encoded by a segment of trajectory in the neural state space. This

allows the neural activity to change within the trajectory (slow manifold), but still maintains the information invariantly[19,37,38].

However, it has been an ongoing debate about which of these two neural activity patterns better characterises short-term memory[22,39]. Through training artificial recurrent neural networks (RNNs), Orhan & Ma found[40] that a wide range of circuit- and task-related factors can affect the emergence of persistent or transient neural activity underlying short-term memory. Nevertheless, if both neural activity patterns can emerge from the task optimization, which implies they are computationally plausible, what are their common underlying neural mechanism? Why does the neural activity appear to be persistent in some situations, but transient in some others? What are the computational principles underlying the neural dynamics? To answer these intriguing and important questions, we need a unified theory to reconcile and explain both the persistent and transient neural activities, which is crucial to provide fundamental insights into the dynamic mechanism and the computational principles of short-term memory, and to facilitate the interpretation of experimental data.

In this study, we trained recurrent neural networks (RNNs) to perform the delayed-response tasks. The stimulus to be memorised was drawn from a prior distribution that can vary from uniform to discretised distribution. This conceptually covered a broad range of cognitive task stimuli, including spatial location[9,21,41,42], colour recognition[43], and object orientation[44]. We showed that the trained RNNs reproduced the typical features of previous experiments[42,45], including persistent activity or transient activity depending on the prior distribution of the training stimulus. We found that both neural activity patterns can be explained by a unified dynamic mechanism: a neural state moving on a low-dimensional manifold driven by a velocity field. We further revealed its computational principle by mapping the RNNs into the noisy-channel model, and showed that the velocity field was shaped by the prior distribution so that the outputs of the RNNs were Bayes-optimal. Overall, our work suggests that in the delayed-response tasks, transient and persistent neural activity might be explained by a unified dynamic model whose dynamics is governed by the Bayesian rule.

Results

The RNN architecture and training

We trained the RNNs (Fig. 1a) to perform the delayed-response task. Each RNN consisted of 12 perception neurons, 1 go neuron, 12 response neurons, and 256 hidden recurrent neurons. Perception and recurrent neurons were entangled with intrinsic Gaussian noise. A task trial has 5 epochs (Fig. 1b). In the perception epoch, the RNNs perceived a stimulus using predefined perception tuning curves (see Fig. 1a and Methods). Stimulus was a random variable defined in $[0°, 360°)$, drawn from a prior distribution consisted of 4 von Mises functions with parameter σ_s to control the width (illustrated in Fig. 1a). After the perception epoch and a short period of delay, go neuron received a go cue signal in the go epoch. Finally, in the response epoch, the RNNs were required to reproduce the previously perceived stimulus using its response neurons. Each RNN was trained in multiple trials with the same prior distribution, but different RNNs could be trained on different prior distributions (with different σ_s, see Methods).

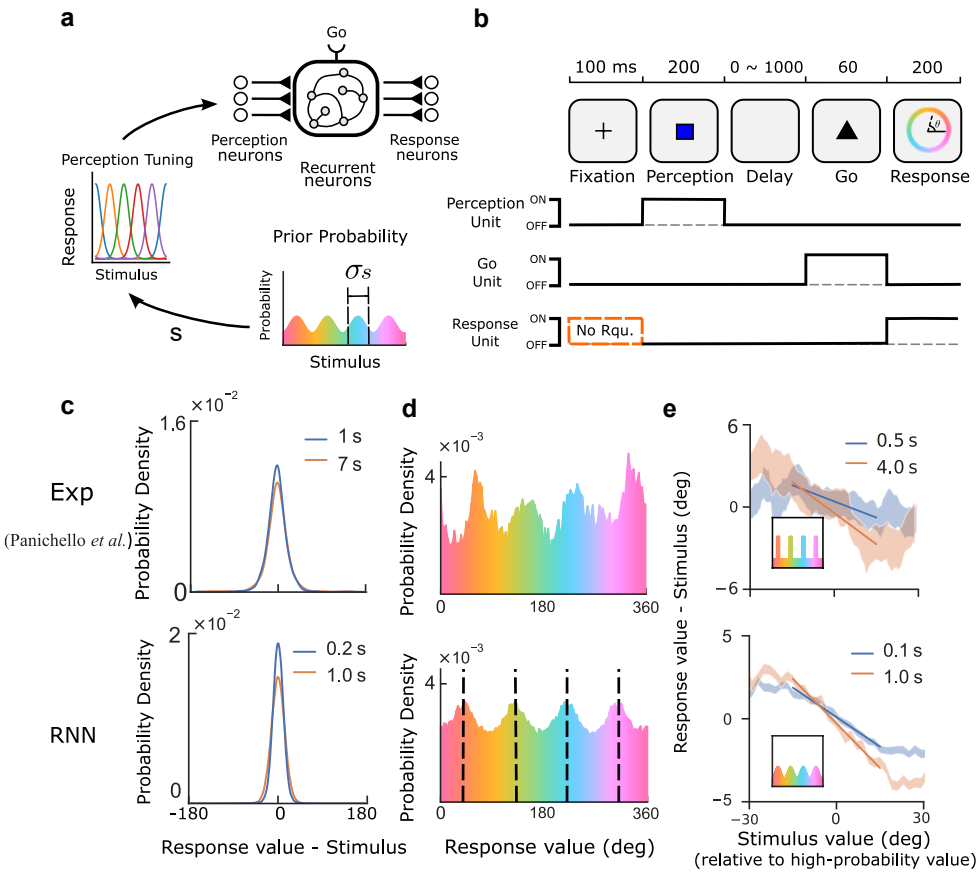

Figure 1. | The architecture, training and behaviour features of the recurrent neural networks (RNNs). (**a**) The RNNs architecture. In each trial, a stimulus was drawn from a prior distribution consists of four von Mises functions with fixed mean and width σ_s (see Methods). Figure shows $\sigma_s = 25°$. Recurrent and perception neurons were entangled with small Gaussian noise. (**b**) Each trial contained five epochs. The RNN was required to memorise the stimulus in the perceptual epoch and reproduce it during the response epoch after a delay. The delay length in different trials followed a uniform distribution from 0 to 1000 ms. The orange dashed rectangle indicates no requirement for the output of the response units. (**c**) Both human participants (upper) and the RNNs (bottom) made larger errors with a longer delay length. (**d**) Both human participants (upper) and the RNNs (bottom) showed preferences for certain values. Black dashed lines indicate high-probability values in the training prior distribution. (**e**) Upper: human participants were trained on a biased prior distribution (inset). Their performance during the last third of the training trials is shown. Bottom: The trained RNNs' results. The error band is the standard error of the mean across trials and RNNs. In (c)(d)(e), 50 RNNs were trained with $\sigma_s = 25°$, and then tested on uniform stimuli. Experimental figures in (c)(d)(e) were plotted using the data published by Panichello *et al.*[45]

The trained RNNs exhibit features of human experiments

The trained RNN reproduced biased behavioural features reported in previous human delayed-response experiments[45,46]. Specifically, the experiments[45] tested human performance on the colour delayed-response task without any pretraining and found that (1) the errors between the response and the stimuli values became larger for longer delay intervals (Fig. 1c) and (2) despite the test stimuli being selected as uniform, the human participants showed preferences for particular stimulus values in the response epoch (Fig. 1d). They hypothesised that these preferred stimulus values reflect the high-probability colours in the biased colour prior distribution in the environment. To test this hypothesis, they trained human participants to perform the delayed-response task with the stimuli drawn from a biased prior distribution with a few high-probability values (Fig. 1e, inset). The human performance in the last third of the training trials is shown in Figure 1e. The negative slopes in the figure indicate that (3) the human participants showed a bias towards high-probability values used in training, and (4) longer delay intervals led to stronger bias (a more negative slope). These two findings suggest that humans can adapt their preferences according to the training context.

Similarly, our trained RNNs also exhibited these four features. First, we trained 50 RNNs with a biased prior distribution (Fig. 1a, $\sigma_s = 25°$) that conceptually mimicked biased colour distribution in the natural environment[47]. Then, we tested the trained RNNs using uniformly distributed stimuli. We found that (1) the error is larger for longer delay (Fig. 1c), as observed in previous experiments; (2) the RNNs showed a preference for certain values (Fig. 1d); (3) these values were exactly the high-probability values (Fig. 1d, e); and (4) longer delay intervals led to stronger bias (Fig. 1e). These results showed that the four behavioural features observed in previous experiments emerged from the trained RNNs.

The trained RNNs exhibit persistent and transient neural activity patterns

In addition to the behavioural features, our trained RNNs showed neural bump activity for all different prior distributions (Fig. 2). Bump activity was found in a previous experiment[42]. To reproduce the bump neural activity in the RNN, we firstly built an RNN decoder to decode the stimulus information value from

the recurrent neural populational firing rates. The RNN decoder sets the recurrent neural populational firing rates as the state at the end of delay epoch, and directly runs the original RNN through the go and response epochs. The reported information value of the RNN during the response epoch is the decoded information value (see Methods). Each neuron's firing rates across different information values were used to construct a tuning curve (SI Fig. 1). For each neuron, the tuning curve was maximised at a preferred stimulus, which was used to label that neuron. Next, we ran one example RNN (for each prior distribution) of one example trial (the stimulus was $180°$). Neural activities are shown in Fig 2a where the neurons are ranked by their labels. The resulting figure shows a bump.

Based on the bump activities, we observed that the RNNs exhibited either persistent or transient neural activity depending on the prior distributions. If the RNN was trained on a uniform or biased prior distribution, the neural bump activity moved slowly during the delay epoch, signifying transient neural activity (Fig. 2a). Five example neural activities (Fig. 2b) also indicated that the RNN exhibited transient neural activity. In contrast, if the RNN was trained on strongly biased prior distributions, the neural activity was persistent after a very quick transient shift of the bump.

We examined in more detail the evolution of the neural activities in the delay epoch based on the neural speed, defined as $v(t) = |\big(\boldsymbol{x}(t+\Delta t) - \boldsymbol{x}(t)\big)|/\Delta t$, where $\boldsymbol{x}(t)$ is a 256-dimensional vector representing the recurrent neural firing rates of the RNN at time t, and $\Delta t = 20$ ms is the computational time interval (see Methods). We trained the RNNs with different prior distributions, and tested them using evenly distributed stimuli. Testing delay length was fixed to 6000 ms to allow the fully decaying behaviour of neural speed (although the RNNs were trained with delay $\sim U(0{,}1000\ \mathrm{ms})$). First, for each RNN, we computed the averaged neural speed across stimuli at every delay time point (Fig. 2c). We found neural speed was dependent on the delay time point. It maintained non-zero values for the entire delay epoch if the prior distribution was weakly biased, but it quickly decayed to zero if the prior distribution was strongly biased. Second, we computed the averaged neural speed across delay time points (excluding the first 200 ms, which may overlap with the transitional period between the perceptual epoch and the delay epoch) for

each RNN under every stimulus. We found the neural speed was dependent on the stimulus values (Fig. 2d). High-probability stimuli led to low neural speed, yet low-probability stimuli led to high speed. Finally, the mean neural speed averaged over the stimuli and delay epoch was dependent on the heterogeneity, σ_s, of the prior distribution (Fig. 2e): speed of strongly biased distribution was smaller than that of a uniform distribution. Overall, the neural speed was dependent on the prior distribution, the delay time, and the stimulus.

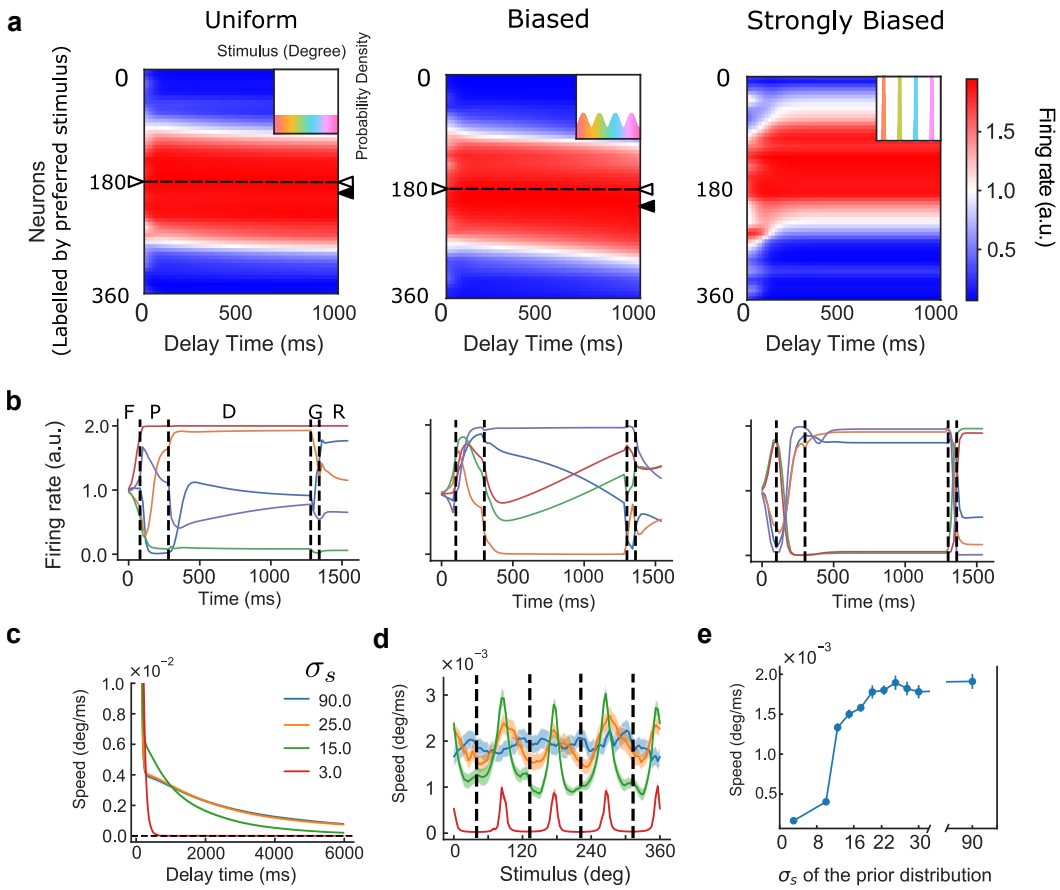

Figure 2. | The trained recurrent neural networks (RNNs) exhibit either transient (left and middle) or persistent (right) neural activity depending on the prior distribution. (**a**) Bump neural activities. Three example RNNs were trained on three different prior distributions (see inset above, $\sigma_s = 90°, 25°, 3°$). Each example RNN was tested on an example trial with a stimulus value of 180°. Neurons were ranked by their preferred stimuli indicated by their tuning curves (see Methods and SI Fig. 1). (**b**) The firing rate time course of five example neurons during a trial. F, fixation; P, perception; D, delay; G, go; R, response. (**c**) Neural speed defines the change of populational firing rate at a unit time. We computed the mean neural speed averaged over different stimuli (100 stimuli evenly sampled from 0° to 360°) for each RNN at every time in the delay epoch. (**d**) The mean neural speed averaged over a delay time window (200 ms, 6000 ms) at different stimuli. Dashed lines indicate high-probability stimuli. (**e**) Mean neural speed averaged over delay windows (200 ms, 6000 ms), and stimuli vs. σ_s of the prior distribution. In (**c**, **d**, **e**), 50 RNNs were trained for each prior distribution, and tested on evenly distributed stimuli and a fixed delay interval of 6000 ms. The error bands and error bars indicating the standard error of the mean across different realizations of RNNs. Error bands of (c) are smaller than the width of the lines. The noise of all RNNs was turned off after training for better visualisation.

Geometric structure of the neural dynamics

We next investigated how and why the RNNs showed persistent or transient neural activity under different prior distributions. We used geometric interpretation to elucidate how the RNN solved the delayed-response task after training. The time-varying neural population activity was represented with a trajectory in an *N*-dimensional space (neural trajectory), where each axis represents one neuron's activation state (*N*=256 in our model). To analyse the dimensionality of the neural activities, we trained 50 RNNs with $\sigma_s = 25°$. Each RNN was tested on 100 trials with evenly distributed stimuli. The corresponding neural activities were recorded. Principal component analysis was used to calculate the cumulative explained variance of different principal components for each RNN (Fig. 3a). The first two principal components explained approximately 90% of the variance of the neural activity, indicating that the neural trajectories were mainly restricted within the two-dimensional plane spanned by the first and second principal components ($PC_1 - PC_2$ plane). As shown in Figure 3b, we projected the neural trajectories of different trials onto a three-dimensional subspace spanned by the first three principal components of the full trials. In the perception epoch, different stimuli in different trials drove the neural states in different directions. In the delay epoch, the neural states converged to a ring-like manifold. In the go and response epochs, the neural states mainly moved along the third principal component, while their projections on the $PC_1 - PC_2$ plane were approximately invariant. These observations suggest that the ring manifold in the $PC_1 - PC_2$ plane encoded memory information for short-term memory.

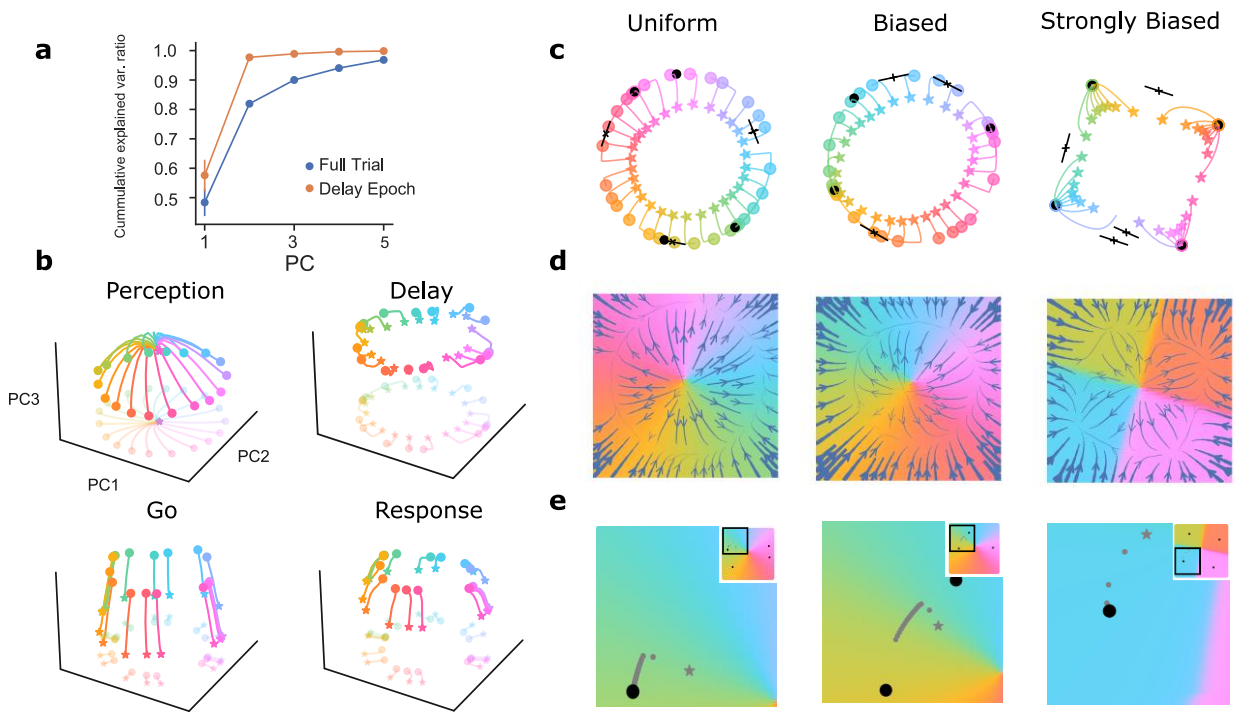

Figure 3. | Geometry of neural dynamics of recurrent neural networks (RNNs) trained on different prior distributions. (**a**) The cumulative explained variance ratios of the principal components ranked according to their eigenvalues from largest to smallest. We trained 50 RNNs with $\sigma_s = 25°$. Each RNN was tested on 100 trials with evenly distributed input stimuli. Error bars (standard deviation across different RNNs) are smaller than the markers. (**b**) Neural trajectories projected onto the three-dimensional space spanned by the first three principal components (darker lines) and their 2D projections (lighter lines). Each trajectory represents one trial, where the stars denote the initial position and the filled circles denote the final position in different epochs. Colour represents the input stimulus in the trial, which was sampled evenly from 0° to 360°. (**c**) The neural trajectories in the delay epoch projected onto the $PC_1 - PC_2$ plane. Fixed points near the heteroclinic cycle are shown. Black filled circles are the attractors and black crosses are the saddles. Each saddle's unstable directions were projected onto the $PC_1 - PC_2$ plane and are indicated by long bars. (**d**) The velocity fields on the $PC_1 - PC_2$ plane. The width of the velocity field lines is proportional to the speed. In the background, the colours represent the decoded information values at the corresponding position (see Methods). (**e**) Example trials show persistent or transient neural activity. The trajectories start from grey stars and approach the attractors (black circles). The length of the delay epoch was 1000 ms, and each

pair of consecutive grey circles represents a fixed time interval of 60 ms. Noises of the RNNs were turned off in testing (see figures with noise in SI Fig. 2)

Different stimuli drive neural states to different angles of a heteroclinic cycle on the $PC_1 - PC_2$ plane

We further studied the ring manifold to characterise the dynamic mechanisms of the RNNs during the delay epoch. From a dynamic point of view, the movement of neural states is driven by fixed points and the velocity field in the state space. We identified fixed points near the ring manifold (Fig. 3c, their largest eigenvalues can be found in SI Fig. 3) and calculated the velocity field of neural states sampled on the $PC_1 - PC_2$ plane (see Methods). The velocity field consisted of 256-dimensional (number of recurrent neurons) vectors, which we projected onto the $PC_1 - PC_2$ plane for visualisation (Fig. 3d). We found that the velocity field was strong both inside and outside the ring, which quickly pushed the neural states towards the ring manifold. On the ring manifold, attractors and saddle fixed points (with unstable directions along the manifold) formed a heteroclinic cycle to generate a slow velocity field. More precisely, if the RNNs were trained on a uniform prior distribution or a biased prior distribution (Fig. 3c, d, left and middle), the heteroclinic cycle was more like a ring, whereas it twisted into a square when the prior distribution was highly biased (Fig. 3c, d, right). Collectively, these analyses indicated that the neural states universally moved towards a heteroclinic cycle under different prior distributions. Different stimuli drove the neural states to different positions on the manifold.

A unified dynamic framework explains persistent and transient neural activities

We computed the neural trajectories of example trials that showed transient or persistent neural activity (Fig. 3e). The information was encoded in a low-dimensional manifold (heteroclinic cycle), on which the velocity field drove the neural state towards different positions encoding different information values (Fig. 3c, d). If the velocity field was strong, the neural states moved quickly and were bounded by the attractors

(Fig. 3e right), and the corresponding neural activity was persistent after a very short period of transition. This scenario matched the discretised attractor model[25]. However, if the velocity field was weak, the neural states moved slowly towards, but did not reach, the attractors during the delay epoch (Fig. 3e left and middle). Hence, the corresponding neural activity was transient. This scenario matched the slow-manifold model[37].

This framework suggests that there is no absolute distinction between transient and persistent neural activity. Neural activity is expected to be persistent after a sufficiently long delay (Fig. 4a) which allows the neural state to reach the attractors. However, if the delay epoch length is short, the same neural activity will be transient. Figure 4b (right panels) shows the neural trajectories of one example RNN (trained on $\sigma_s = 3°$) with a short delay, in which the fast transition of the neural states leaked out to the go and response epochs. Therefore, the movement of neural states was a continuous process across different epochs. These results suggested that transient and persistent neural activities are in fact the same dynamic process, but appear to be different due to the different conditions of the experimental measurements.

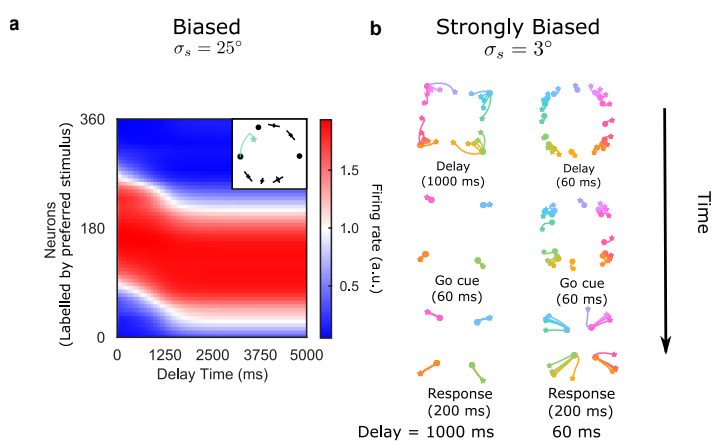

Figure 4. | Transient and persistent neural activities are relative. (a) Both neural activities can be explained by a neural state that moves in a manifold. If the delay is too long, the neural state reaches the attractors (inset) and becomes persistent, while being transient within the first 1000 ms. **(b)** If the delay is too short, transient movement along the angular direction may also occur in the go and response epochs. The example RNN was trained with a delay length $\sim U(0, 1000\text{ ms})$. The $PC_1 - PC_2$ plane was obtained by fitting the neural trajectories in the delay (20 trials with evenly distributed stimuli).

The RNNs use the angle of the heteroclinic circle to encode information

Next, we determined how the information was encoded in the low-dimensional manifold (heteroclinic cycle). Figure 3b presents a hypothesis that the information is encoded by the angle of the heteroclinic cycle. To test this hypothesis, we used the RNN decoder to decode a grid mesh of neural states on the $PC_1 - PC_2$ plane to their corresponding information values. We found that the information values were encoded along the angular direction, but not the radial direction (Fig. 3d, background colour). Quantitatively, the angle of the $PC_1 - PC_2$ plane (ϕ) and the information value (θ) had a smooth functional relationship (Fig. 5a). We also calculated its derivative, $d\phi/d\theta$, which measures the angular volume used to encode a unit of information. If the RNNs were trained on a uniform prior distribution, similar angular volumes were used to encode different information values, but if the prior distribution was biased or strongly biased, larger angular volumes were used to encode the high-probability information values (Fig. 3d, 5b).

The distribution of attractors is consistent with the training prior distribution

As information was encoded by the angles, the positions of the attractors were critical. From a dynamic point of view, the neural states during the delay epoch tended to move towards the attractors. At the level of behaviour, the RNNs tended to report information values that were encoded by either attractors or neural states nearby. To test this idea, we decoded the attractors' information values for different independently trained RNNs using the RNN decoder (Fig. 5c). We found that if the RNNs were trained on a uniform prior distribution, the attractor information values were approximately uniformly distributed. Therefore, on average, the RNNs had no preference for any information value. In contrast, with a biased or strongly biased prior distribution, the attractors tended to accumulate at high-probability information values. Hence, the RNNs preferred to report high-probability information values, which is consistent with the results of previous experiments (see Fig. 2b, c).

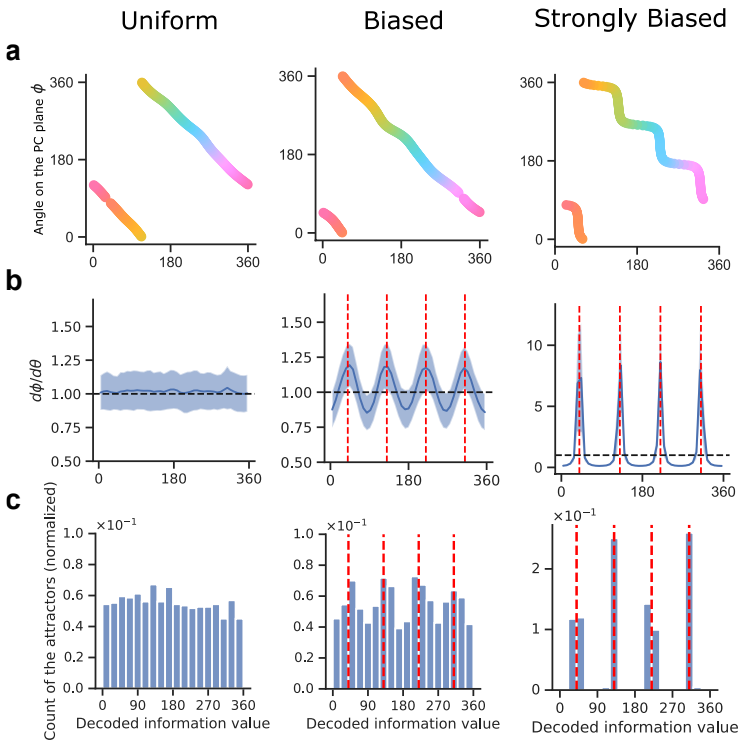

Figure 5 | Encoding schemes of the recurrent neural networks (RNNs). (**a**) The relationship between the angle, ϕ, on the $PC_1 - PC_2$ plane and the decoded information value, θ, of one RNN for each prior distribution (see Methods). (**b**) The derivative of the angle, ϕ, as a function of the decoded value, θ. The shaded area represents the standard error across realisations of RNNs. Error bands indicate the standard error of the mean across different RNNs. (**c**) Distribution of the decoded values of attractors (bin size = 20°). In **b** and **c**, 250 RNNs were trained independently for each prior distribution. Red dashed lines indicate the high-probability values in the prior distribution.

A diffusion–drift model maps the dynamic properties of the RNNs

Geometric differences between the dynamics of the RNNs trained on different prior distributions resulted in different neural activities, persistent or transient. However, it remained unclear why and how the prior distributions affected these dynamics. To unravel the underlying computational principle, we mapped the dynamic properties of RNNs in the delay epoch to a diffusion–drift model (DDM, see Methods)[30]. The results suggested that the information change during the delay epoch was subject to two factors: diffusion, as the information randomly diffused, leading to unbiased errors; drift, as the information systematically changed, leading to biased errors. These two processes conceptually matched our geometric pictures.

Diffusion is due to the intrinsic noise added to the RNN, and drift is due to the systematic velocity field on the information manifold. We fitted the DDM to an example RNN ($\sigma_s = 25°$), in which the drift force and noise strength were free parameters (see Methods). The fitting results shown in Figure 6a suggest that the DDM was sufficient as a simplified information-processing model of the RNN.

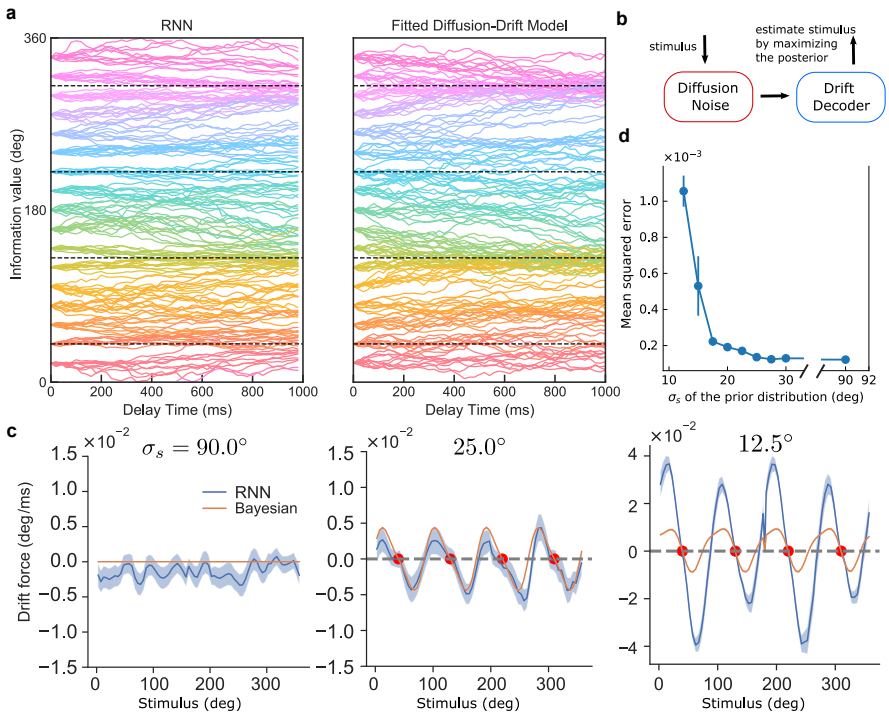

Figure 6 | Computational principle of Bayesian inference in the recurrent neural networks (RNNs). (**a**) The simulated evolution of information values during delay epoch. Left: We ran a trained RNN ($\sigma_s = 25°$) with multiple trials using different stimuli. The neural states during the delay were decoded to information values using the RNN decoder. Each line is one trial and the colour represents the stimulus value at the beginning of the delay epoch. Right: We fitted the diffusion strength and drift force of a diffusion–drift model (DDM) to the left RNN, and then simulated the DDM numerically (see Method). Dashed lines indicate high-probability information values. (**b**) If the drift force was weak, the DDM was simplified as a noisy-channel model (see Method). Stimulus information was corrupted by the diffusion process, then decoded by the drift process. The drift force that maximised the posterior distribution is called the Bayesian-optimised force. (**c**) Comparisons of the drift force of the RNNs with the Bayesian-optimised force for different prior distributions. Red dots indicate the high-probability information values. We trained 50 RNNs

for each prior distribution. For a given prior distribution, each RNN was fitted to a DDM to obtain the fitted drift force and noise strength. The mean of 50 drift forces is shown as the blue line. Meanwhile, based on the noise strength of the 50 fitted DDM, 50 Bayesian-optimised forces were obtained (see Methods). Orange lines indicate the average. (**d**) Mean squared error between the mean RNN drift force (blue lines in (**c**)) and the mean Bayesian-optimised force (orange lines in (**c**)). Error bands and error bars represent standard error of the mean across 50 realizations of RNNs. Error bands of Bayesian-optimised forces are too narrow to be visible.

The drift force of the RNN is Bayesian-optimised

Under conditions in which the drift force was weak, we further simplified the fitted DDM to a noisy-channel model (Fig. 6b, see Methods). Consequently, we proposed the following interpretation of how the prior distribution affects the RNN's velocity field, which has been simplified as the drift force of the DDM. The RNN senses a stimulus in the perceptual epoch and stores it in the form of a populational neural firing pattern. This neural firing pattern is a neural state in a low-dimensional manifold according to the above geometric analysis. During the delay epoch, the neural state unavoidably deviates from its original position due to intrinsic noise within the RNN (diffusion in the DDM), which leads to a short-term memory error. This process can be summarised as the stimulus passing through a noisy channel and providing noisy information output. The noisy information is then received by a drift decoder, which infers the original stimulus based on the noisy information received and prior knowledge about the prior distribution (acquired during the training of the RNNs). The inference result depends on the drift force. We hypothesised that the drift decoder is a Bayesian-optimised decoder, and its corresponding drift force is named the Bayesian-optimised force.

We tested whether the force fields in the RNNs performed Bayesian inference, i.e., whether the drift forces fitted from the RNNs were Bayesian-optimised. Specifically, we fitted a DDM for each RNN and obtained the drift force, $f(\theta)$, and the noise strength, ν, where θ is the information value from $0°$ to $360°$. The fitted noise strengths were set to be equivalent to the corresponding parameters in the noisy channel.

We then used gradient ascent to maximise the averaged posterior distribution (see equation [22] in the Methods) and obtained the Bayesian-optimised force, $f_{\mathrm{bay}}(\theta)$. A comparison of $f_{\mathrm{bay}}(\theta)$ and $f(\theta)$ in Figure 6c shows that they agreed with each other well when the drift force was weak or θ is within the neighbourhoods of the attractors. The mean squared error between the RNN drift force and the Bayesian-optimised force (Fig. 6d) was small when the prior distribution was weakly biased (large σ_s) which also indicated that the drift force was weak. This result accords with the assumption in our theory. Overall, our analyses suggested that the RNN generates velocity fields such that the change in the neural state is equivalent to performing the maximum a posteriori estimation. Bayesian inference is a key computational principle underlying the dynamic properties of the trained RNN, while different dynamic properties lead to different phenomena, including both persistent and transient neural activities.

Discussion

We trained RNNs to study the unified neural mechanism for short-term memory associated with persistent or transient activities. We found that the trained RNNs reproduced several features in the experiments, including persistent and transient neural activity depending on the training prior distribution. Reverse-engineering revealed a unified model that can explain both persistent and transient neural activity: information is encoded in a neural state moving on a low-dimensional manifold driven by a velocity field whose strength is adjusted properly to reduce the memory error by maximizing the posterior distribution.

We developed an RNN that can reproduce experimental features. Specially, in the delayed-response task, subjects are biased when reporting certain values[45,48], which is thought due to the adaptation to natural statistics[49,50]. Our trained RNNs reproducing the biased reporting behaviour thus complement the data from animal experiments to help elucidate how and why the memory bias occurs, and are ready for further numerical experiments.

By analysing the RNN, we proposed a unified dynamic model that conceptually unifies two previously proposed theories of short-term memory: the discretised attractor model explaining persistent neural activity[25] and the slow-dynamic model of transient neural activity[37]. Our findings also suggest a new perspective in reconciling the heated debate between persistent and transient neural activity: they are not fundamentally distinct phenomena, but just specific configurations (specifically, velocity field strength) of the same unified framework. In this framework, heteroclinic orbits have different attractor locations and velocity fields. If the velocity field is weak, the neural states move slowly, which corresponds to transient activity. Yet, strong velocity field would quickly stabilises the neural states to the discretised attractors, and results in persistent neural activity. More generally, there should be a transient time window based on the strength of the velocity field, with stronger velocity fields having shorter transient time windows. In experimental systems, if the length of the delay is smaller than the transient time window, neural activity appears to be transient; otherwise, it appears to be persistent. Based on this unified explanation, a better description of neural activity might be the transient speed and time window instead of a simple categorisation as being persistent or not. We suggest that future research focus on how the transition speed and time window are affected by various factors involved in working memory.

The aforementioned unified dynamic model is partially supported by the results of previous experiments. Murray *et al.* trained[14] monkeys to perform the oculomotor delayed-response task, for which the stimulus is spatial orientation. They found that neurons in the prefrontal cortex use a stable ring-shaped manifold to encode spatial information, similar to the scenario presented in Figure 3. Panichello *et al*. trained[45] human to perform the colour delayed-response task, where a colour is drawn from a biased distribution. After training, the subjects were more likely to report values with high probability in the biased distribution. This phenomenon can be explained by our unified model. Specifically, after training on a biased prior distribution, the human brain is more likely to use attractors to encode the high-probability values (Fig. 5). These attractors attract neural states during the delay epoch. Consequently, subjects tend to report high-probability values after training. Finally, similar to the case of a strongly biased prior distribution of discretised stimulus values, Inagaki *et al*. trained[51] mice to perform the delayed-response task with two

types of music, leading to a left lick and a right lick in the response epoch. They found that neurons in the mouse anterior lateral motor cortex form discrete attractors and thus, generate persistent neural activity during the delay epoch. These results are consistent with and can be unified by our model. To further test our theory, one can train animals to perform the oculomotor-delayed-response task with different prior distributions. The resulting neural dynamics should also be different, including the transient speed, neural bump position etc.

The unified dynamic model provides a bottom-up explanation of the transient neural trajectory, but it remains unknown why neural activity changes during the delay. This phenomenon may have multiple functional implications. First, a slow manifold-encoding mechanism is more robust to noise than a discretised fixed-point model. Ghazizadeh and Ching trained RNNs on a working memory task[37], and found that RNNs showing transient neural activity were more robust to noise than those with persistent neural activity. One possible origin of such robustness is untangling. Tangling is defined as two very closed neural states resulting in very different future behaviours. In our trained RNNs, the information was encoded in the angular direction. However, the neural states moved along the radial direction during perception epoch and the early stage of the delay (Fig. 3). This process increases the distance between two adjacent neural states with similar information, hence reducing information tangling and enhancing memory robustness[52]. The second possible reason for transient neural activity may be the change of latent information. For example, time information was suggested to be dynamically encoded[53]. Harvey et al. suggested the dynamic activity of posterior parietal cortex neurons reflects spatial or temporal progression through the task[20].

Besides robustness, our work implies one additional functional implication of transient neural activity: it allows the neural network to manipulate information during the delay. Over the last several decades, neural activity during the delay was thought to maintain information[7,9,15]. However, in recent years, evidence has increasingly suggested that neural activity also plays a crucial role in manipulating information[22,54]. For example, subjects participating in delayed-response tasks are biased to report particular values. The bias effect increases as the delay epoch increases, which implies the constant changing of information during the delay[45]. However, how and why neural activity manipulates the information is

unclear. Here, we showed that the manipulation of information is achieved by the movement of neural states along the angular direction, guided by the heteroclinic orbit during the delay; hence, it is directly related to transient neural activity. This is supported by previous experiments where the change of behaviour was correlated with the change of neural bump position[42]. Furthermore, why did the RNNs manipulate information during delay? Our noisy-channel model suggests that the information is manipulated to reduce the effect of intrinsic noise and thus, achieve the maximum of the posterior estimation. Therefore, our results are consistent with the idea that a neural system adapts to natural statistics, especially in working memory and decision-making tasks[49,55–59].

The conclusions of this study are based on RNNs trained on delayed-response tasks. Whether these findings can be generalised to other working memory tasks warrants further investigation. In future studies, we expect more dynamic and statistical analyses to be applied to RNN models trained for various working memory tasks. In summary, we reverse-engineered RNNs and provided a unified perspective to reconcile persistent and transient neural activity, and suggested a possible underlying computational principles.

Methods

RNN architecture

The RNN consists of 12 perception input neurons, 1 go input neuron, 12 response output neurons and 256 fully connected recurrent neurons (Fig. 1a). The dynamic equation of the RNN is[38]

$$\boldsymbol{x}_t = (1-\alpha)\boldsymbol{x}_{t-1} + \alpha\left(\boldsymbol{W}^{rec}\boldsymbol{r}_{t-1}^{rec} + \boldsymbol{W}^{in}\boldsymbol{u}_t + \sqrt{2\alpha^{-1}\sigma_{rec}^2}\boldsymbol{\epsilon} + \boldsymbol{b}\right), \quad (1)$$

where $\boldsymbol{x}_t$ is a 256-dimensional vector representing the activities/state of recurrent neurons (neural state) at time step t, and $\alpha = \Delta t / \tau$, where Δt is the time length for each computational step, τ is the neuron membrane time constant. In this study, we set $\Delta t = 20$ ms and $\alpha = 1$, similarly as the scheme used in previous work[38,40,60]. The change of neural state subjects to 4 factors: 1) The inputs from the other recurrent neurons, $\boldsymbol{W}^{rec}\boldsymbol{r}_{t-1}^{rec}$, where $\boldsymbol{r}_t^{rec} = \tanh(\boldsymbol{x}_t)$ and $\boldsymbol{W}^{rec}$ is the connection weights between the recurrent

neurons with diagonal elements fixed to 0 (self-connections are not allowed). The neural firing rate is defined as $\boldsymbol{f}_t^{rec} = 1 + \boldsymbol{r}_t^{rec}$, where adding 1 makes sure the firing rate is positive. 2) The inputs are from perception and go neurons, $\boldsymbol{W}^{in}\boldsymbol{u}_t$, where $\boldsymbol{u}_t$ consists of 12 components for the perception neurons and 1 component for the go neuron, and $\boldsymbol{W}^{in}$ is the input weights. 3) The intrinsic noise in the RNN, $\sqrt{2\alpha^{-1}\sigma_{rec}^2}\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon}$ is a 256-dimensional noisy vector drawn from the standard normal distribution at each time step, and $\sigma_{rec} = 0.2$ is the noise strength. 4) A constant bias current $\boldsymbol{b}$.

The activities of the 12 response output neurons is read from the $\boldsymbol{r}_t^{rec}$,

$$\boldsymbol{z_t} = \boldsymbol{W}^{out}\boldsymbol{r}_t^{rec} + \boldsymbol{b}^{out}. \tag{2}$$

RNN training

In the beginning of each trial, the recurrent neural state is initialized to be a very small gaussian noise. In the fixation epoch, the RNN receives no inputs (i.e., $\boldsymbol{u_t} = 0$), and there's no requirement on the response neurons (i.e., target output $\boldsymbol{z_t}$ is arbitrary). In the perception epoch, a stimulus θ_s is drawn from a prior distribution,

$$P_s(\theta_s) = \frac{1}{4}\sum_i \mathcal{VM}(\theta_s - \mu_i; \sigma_s^2), \tag{3}$$

which is the sum of four von Mises functions (circular version of the normal distribution) with mean values at $\mu_i \in \{40°, 130°, 220°, 310°\}$,

$$\mathcal{VM}(\theta_s - \mu_i; \sigma_s^2) = \frac{1}{2\pi I_0(1/\sigma_s^2)}\exp[\cos(\theta_s - \mu_i)/\sigma_s^2], \tag{4}$$

where σ_s is a free parameter to control the bias of the prior distribution, and $I_0(\cdot)$ is the modified Bessel function of order 0 for normalization purpose. Units of θ, μ and σ are converted from degree to rad before feeding into the von Mises function. The stimulus drawn from the above prior distribution is sensed by the 12 perception neurons, whose tuning curves are also von Mises functions $\mathcal{VM}\left(\theta_s - \mu_{p,i}; {\sigma_p}^2\right)$ with $\mu_{p,i} = \frac{360°}{12}i$ and $\sigma_p = \frac{1}{2}\frac{360°}{12}$ for $i = 0, 1, \ldots, 11$ (Fig. 1a). Therefore, in the perception epoch, the perception

components of the input vector $\boldsymbol{u}_t$ are $u_{t,i} = \mathcal{VM}\left(\theta_s - \mu_{p,i}; {\sigma_p}^2\right)$ and the go component is 0 ($u_{t,12} = 0$). In this epoch, we required the outputs to be 0 (i.e., $\boldsymbol{z}_t = 0$). Next, during the delay epoch, all inputs and outputs neurons are 0. The delay length is drawn from an uniform distribution U(0 ms, 1000 ms) in each trial. Then, in the go epoch, we set $u_{t,12} = 1$ to indicate a go cue and the rest of inputs and the outputs are 0. Finally, in the response epoch, the 12 response neurons need to reproduce the neural activity of the perception neurons in the perception epoch, i.e. $\boldsymbol{z}_{t,i} = \mathcal{VM}\left(\theta_s - \mu_{p,i}; {\sigma_p}^2\right)$ for $i = 0, 1, \ldots, 11$. The actual output $\boldsymbol{z}_t$ in the response epoch is then decoded into a report information value using the populational vector method[38,42]

$$\hat{\theta} = \arg\left(\sum_{m=1}^{N=12} \bar{z}_m \, e^{i\mu_{p,m}}\right), \tag{5}$$

where arg means angle, and $\bar{z}_m$ is the averaged neural activity of the response neuron m (mth component of vector $\boldsymbol{z}_t$) within the middle response time window (from 60 to 140 ms in the response epoch). Equation (5) is defined as the report information value of the RNN.

The above statements define the output of the RNN. The corresponding loss function is

$$Loss = \frac{1}{T_{tot}} \sum_t m_t \left[|\boldsymbol{z_t} - \tilde{\boldsymbol{z}}_t|^2 + \frac{1}{N_{rec}} \left(\beta T |\boldsymbol{W}_{rec}|^2 + \gamma |\boldsymbol{r}_{rec}(x) \; + \; 1|^2\right) \right], \tag{6}$$

where m_t is the mask that is 0 in the fixation epoch and 1 in the rest epochs, $\tilde{\boldsymbol{z}}_t$ is the target response output as stated above, T_{tot} is the trial time length, N_{rec} is the number of recurrent neurons, β and γ in the regularization terms constrain the recurrent weights and neural firing rates. This loss function is minimized by gradient descent using the Adams optimizer[61].

However, since directly training the RNN to optimize the above loss function is difficult, we employ the progressive training protocol[19]: First, we train the RNN with no noise, no regularization terms, uniform prior distribution, and delay epoch length fixed to 0. Secondly, we retrain the RNN with delay epoch length drawn from U(0 ms, 1000 ms). Thirdly, the RNN is retrained considering the noise and regularizations. Finally, we retrained the RNN with the desired prior distribution.

Decoding the Neural States

We built an RNN decoder to decode a neural state $\boldsymbol{x}_t^{rec}$ to its corresponding stimulus information value. Specifically, an RNN decoder is the same as the original RNN trained on the task, but its neural state is initialized as the to-be-decoded neural state. Then, the RNN decoder runs directly through the go and response epoch. The output information value in the response epoch is the decoded information value of the to-be-decoded neural state.

We used the RNN decoder to find out the relationship between angles on the heteroclinic cycle and the stimulus information values. We ran an example RNN with multiple trials with different stimuli. At the end of delay epoch (delay length fixed to 800 ms), the neural state should locate near the heteroclinic cycle. Therefore, we collected these neural states from different trials, and projected them onto the $PC_1 - PC_2$ plane. Their averaged projected vector norm defines the radius. Using this radius, we sampled new neural states on a ring with different angles ϕ. Since the new neural states are represented by their 2D projections, we applied inversed PCA to recover them back to the 256-dimensional state space. Then the recovered neural states are decoded by the RNN decoder. Collectively, using this method we can obtain the stimulus information value encoded by the angle of the heteroclinic cycle (Fig. 5).

Besides the neural states on the heteroclinic cycle, we also sampled a grid of neural states on the $PC_1 - PC_2$ plane using the above method, which yields the landscape of stimulus information values encoded by a wide range of the $PC_1 - PC_2$ plane (Fig. 3).

Neural bump activity

First, we labelled each recurrent neuron with its preferred information value, which is defined as the maximum point of its tuning curve. In order to measure the tuning curve, we ran the RNN with 360 stimuli that are equally spaced from 0 to 360 degree and collected the neural states $\boldsymbol{x}_i^{rec}(t)$ during the whole delay epoch, where i indicates the ith stimulus. At a specific delay time t, each neural state $\boldsymbol{x}_i^{rec}(t)$ can be decoded to an information value by the RNN decoder. Therefore, we have (number of stimuli × number of

delay time points) pairs of (information value, neural state), which indicate the information value -- neural state relationship. The jth component of a neural state, up to a nonlinear function (tanh + 1), is the firing rate of the jth neuron. Therefore, we obtained the information value – firing rate relationship, which is the tuning curve (shown in SI Fig. 1). Its maximum point is the preferred information value used to label the corresponding neuron.

After labelling the neurons, we can draw out the neural bump activity. We ran the RNN with one trial (stimulus = $180°$) and collect the neural firing rates during the delay. Neurons with similar labels are binned into neural units (bin size = $5°$) and the firing rate of each neural unit is defined as the average of the individual neuron firing rate within the bin. Then the neural units are ranked in the order of their labels to obtain the neural bump activity (see Fig. 2a).

Identifying velocity field and fixed points

The dynamic equation of the RNN, Eq. (1), can be reformulated in the form of

$$\boldsymbol{x}_t - \boldsymbol{x}_{t-1} = F(\boldsymbol{x}_{t-1}), \tag{7}$$

which defines the velocity field $\boldsymbol{v}(\boldsymbol{x}) = F(\boldsymbol{x}) \,/\, \Delta t$. In this study, we set $\Delta t = 20$ ms. The global minimum points of speed $v(\boldsymbol{x}) = |\boldsymbol{v}(\boldsymbol{x})| = 0$ are the fixed points. Specifically, we ran the trained RNN with 500 trials with stimuli equally distributed from $0°$ to $360°$. At the beginning of the delay epoch in these trials, neural states should locate near the heteroclinic cycle approximately on the 2D space, hence they are set to be the initial searching points. Then, starting from these initial neural states, we searched for the global minima of speed using the gradient descent, where the independent variable is the position $\boldsymbol{x}$. Due to the imperfection of the gradient descent algorithm, we may end up with local minima. Nevertheless, these local minimum points are similar to the fixed points[62], hence they are also called fixed points in this paper. For each fixed point, we computed the eigenvalues and eigenvectors of its Jacobian matrix $J = \frac{\partial F(\boldsymbol{x})}{\partial \boldsymbol{x}}$. A fixed point is called an attractor if its largest eigenvalue is negative, while it is called saddle point if some

eigenvalues are positive. The eigenvectors of the positive eigenvalues are projected and shown on the $PC_1 - PC_2$ plane (Fig. 3). Only fixed points near the heteroclinic cycle are shown.

Fitting the Diffusion-Drift model (DDM) to the RNN

The DDM is described by

$$d\theta(t) = f\big(\theta(t)\big)dt + \nu d\xi(t), \tag{8}$$

where $\theta(t)$ is information value restricted in a periodic domain $\theta(t) \in [0,360)^\circ$, $f(\theta)$ is the drift force, $\nu d\xi(t)$ is a gaussian noise process with noise strength ν. This DDM decomposed the change of information during the delay epoch into two processes: diffusion ($\nu d\xi(t)$) and drift ($f(\theta)dt$). These two processes correspond to two features of our RNN. Specifically, diffusion is due to the intrinsic noise of neurons in the RNN, and drift is a systematic change of information, which is induced by the velocity field of the RNN.

The DDM describes the information changes during the delay. Similarly, we found that the decoded information value of the neural state also changes during the delay epoch in the RNN (Fig. 6). Therefore, the goal is to find proper drift force and diffusion strength so that the results of the DDM can best match the results of the RNN. The fitting procedure contains two steps.

The first step is fitting the drift force $f(\theta)$ in Eq. (8). We ran the trained RNN through multiple trials with the intrinsic noise turned off ($\sigma_{rec} = 0$ in Eq. (1)). The neural states are collected during the delay epoch and decoded using the RNN decoder, which yields the information values $\theta(t)$ during the delay epoch. Because the noise was turned off, we assumed that the change of θ during the delay epoch was only due to the drift force. By removing the diffusion term of the DDM (Eq. (8)), only the drift force is left,

$$d\theta(t) = f(\theta)dt. \tag{9}$$

By discretising the time, we obtained the drift force

$$f(\theta) = [\theta(t + \Delta t) - \theta(t)]/\Delta t. \tag{10}$$

However, the DDM is not a perfect description of the RNN, and in some cases, despite that $\theta(t_1) = \theta(t_2)$, $\theta(t_1 + \Delta t) \neq \theta(t_2 + \Delta t)$, hence the resulting forces computed from Eq. (10) are different. To resolve this issue, we took the average of the forces with similar information values (bin size $\Delta = 5°$), i.e., $f(\theta) = \frac{1}{N}\sum_{|\theta(t_i)-\theta|<\Delta/2} f\big(\theta(t_i)\big)$.

The second step is fitting the diffusion noise ν. We ran the trained RNN with multiple trials with the noise turned on. For each trial, we decoded the neural states at the beginning of the delay epoch, θ_0, and at the end of the delay epoch, θ_T. Hence, we obtained a three-dimensional sample (θ_T, θ_0, T) for each trial, where T is the delay epoch length. Due to the noise, θ_T is not deterministic given θ_0 and T. Instead, it should follow a conditional distribution $P_\nu(\theta_T|\theta_0, T; \nu)$ which relies on the parameter ν.

We calculated the conditional distribution by solving the DDM. Specifically, the DDM can be rewritten into a Fokker-Planck equation

$$\frac{\partial}{\partial t}P(\theta, t) = -\frac{\partial}{\partial \theta}f(\theta)P(\theta, t) + \frac{\nu^2}{2}\frac{\partial^2}{\partial^2\theta}P(\theta, t), \tag{11}$$

with the initial condition $P(\theta, 0)$ as a very narrow von Mises function cantered on θ_0. This is used to indicate that the delay epoch starts exactly at θ_0. The solution of this Fokker-Planck equation gives the probability distribution of θ_T at the end of the delay epoch, i.e., $P_\nu(\theta_T|\theta_0, T; \nu) \equiv P(\theta = \theta_T, T)$, given the initial condition θ_0 and diffusion strength ν.

We solved the Fokker-Planck equation numerically. We discretized $\theta \in [0°, 360°]$ into $\theta_i = i\Delta\theta$, where i is an integer, $\Delta\theta = 5°$. Consequently, the Fokker-Planck equation becomes a system of linear differential equations, and can be solved analytically[45].

Given the samples (θ_T, θ_0, T) generated by the RNN and the conditional distribution $P_\nu(\theta_T|\theta_0, T; \nu)$ computed from the DDM, we use the maximum likelihood estimation to fit ν. The negative likelihood function is defined as

$$L(\nu) = \frac{-1}{N}\sum_{j=1}^{N} \ln\, P_\nu\big(\theta_{T,j} \,\big|\, \theta_{0,j}, T_j;\ \nu\big), \tag{12}$$

where N is the number of samples. It is minimized with respect to parameter ν by the Brent's minimization method.

Simplify the DDM into a noisy-channel model and obtain Bayesian-optimized force

We built a noisy-channel model to relate the training prior distribution $P_s(\theta_s)$ to the drift force $f(\theta)$ under the assumption that the drift force is weak. Under this assumption, we can solve the DDM (Eq. (8)) using the perturbation theory. Specifically, the drift force $f\big(\theta(t)\big)$ is considered as the first order small number, noise term $\nu d\xi(\mathrm{t})$ is the zeroth order, and the information value $\theta(t)$ can be approximated by a linear expansion to the first order, i.e., $\theta(t) = \tilde{\theta}(t) + \theta^{(1)}(t)$, where $\tilde{\theta}$ is the zeroth order term and $\theta^{(1)}$ is the first order term. Inserting them into the DDM Eq. (8), the zeroth order yields

$$d\tilde{\theta}(t) = \nu d\xi(\mathrm{t}), \tag{13}$$

with initial condition $\tilde{\theta}(0) = \theta_s$. Here we ignored the noise within the perception epoch, hence θ_s is exactly the stimulus drawn from the prior distribution $P_s(\theta_s)$ (Eq. (3)). The solution of this equation at the end of the delay epoch is approximately a von Mises function (Eq. (4)),

$$P_{n|s}\big(\tilde{\theta}(T) \,\big|\, \theta_s; T\big) = \mathcal{VM}\big(\tilde{\theta}(T) - \theta_s, \nu^2 T\big), \tag{14}$$

The delay length T is drawn from a uniform distribution $P_t(T) = U(0, 1000)$ ms.

Similarly, the first order perturbation yields the drift equation

$$d\theta^{(1)}(t) = f\big(\theta^{(1)}\big)dt, \tag{15}$$

with initial condition $\theta^{(1)}(0) = 0$. To help interpretating this equation, we shift the variable by a constant, i.e., $\theta_f(t) = \theta^{(1)}(t) + \tilde{\theta}(T)$. This shifting will not change the form of Eq. (15) since the left-hand side is a differential operator, which yields

$$d\theta_f(t) = f\big(\theta_f\big)dt, \tag{16}$$

with the initial condition $\theta_f(0) = \tilde{\theta}(T)$. The solution of this equation at T is $\theta_f(T) = \tilde{\theta}(T) + \theta^{(1)}(T) \equiv \theta(T)$, which is the reported stimulus information value at the end of the delay epoch. Collectively, in order to obtain the solution of Eq. (8), we need to use initial condition θ_s to compute $\tilde{\theta}(T)$ (Eq. (14)), and then use $\tilde{\theta}(T)$ as the initial condition to compute the reported information value $\theta_f(T)$ (Eq. (16)).

These two equations have physical meanings. The stimulus θ_s first undergoes diffusion process and becomes a noisy state $\tilde{\theta}(T)$ due to the intrinsic noise in the RNN. Then the diffusion error in $\tilde{\theta}(T)$ is corrected by the drift equation. One way to correct error is the maximum a posteriori estimation. Specifically, the posterior can be obtained using the Bayes' theorem

$$P_{s|n}\left(\theta_s \,\middle|\, \tilde{\theta}(T), T\right) = \frac{P_{n|s}\left(\tilde{\theta}(T) \,\middle|\, \theta_s, T\right) P_s(\theta_s)}{\int P_{n|s}\left(\tilde{\theta}(T) \,\middle|\, \theta_s, T\right) P_s(\theta_s)\, d\theta_s}, \tag{17}$$

where we used the condition that θ_s and T are independent. The goal of the drift equation is using the drift force to infer the original information θ_s based on the input noisy information $\tilde{\theta}(T)$. Therefore, we defined a score function to evaluate the performance of the drift force as the average performance of all possible noisy information $\tilde{\theta}(T)$ and delay length T,

$$\mathrm{Score}(f) = \iint P_{s|n}\left(\theta_s = \theta_f(T) \,\middle|\, \tilde{\theta}(T), T\right) P_{nt}\left(\tilde{\theta}(T), T\right) dT d\tilde{\theta}, \tag{18}$$

where the joint distribution is

$$P_{nt}\left(\tilde{\theta}(T), T\right) = \int P_{n|s}\left(\tilde{\theta}(T) \,\middle|\, \theta_s, T\right) P_s(\theta_s) P_t(T)\ d\theta_s. \tag{19}$$

This score is maximized by the Bayesian-optimized force

$$f_{\mathrm{bay}} = \arg\max_f \mathrm{Score}(f). \tag{20}$$

In order to compute the Bayesian-optimized force numerically, we firstly parameterized the drift force using the Fourier Series

$$f\left(\theta_f\right) = \sum_{i=0}^{M} a_i \sin\left(i\theta_f\right) + \sum_{i=0}^{M} b_i \cos\left(i\theta_f\right), \tag{21}$$

where the parameters are $\omega = \{a_i \mid i = 1,2,\dots,M\} \cup \{b_i \mid i = 1,2,\dots,M\}$. In practice, we included the Fourier terms up to $M = 10$.

Next, we computed the score function. Firstly, a batch of N samples $\left(\tilde{\theta}_i(T_i), T_i\right)$ are drawn from the joint distribution $P_{nt}\left(\tilde{\theta}(T), T\right)$ (equation (19)). Secondly, using $\tilde{\theta}_i(T_i)$ as the initial condition, $\theta_{f,i}(T_i)$ can be obtained by solving Eq. (16) using the Euler method. Finally, the integral in the score function Eq. (19) can be approximated by the Monte Calo method

$$\mathrm{Score}(f) = \frac{1}{N}\sum_{i=0}^{N-1} P_{s|n}\left(\theta_s = \theta_{f,i}(T_i) \,\middle|\, \tilde{\theta}_i(T_i), T_i\right), \tag{22}$$

After obtaining the score function, we updated the drift force once using the gradient ascent

$$\omega = \omega + \delta\frac{\partial\,\mathrm{Score}(f)}{\partial\omega}, \tag{23}$$

where δ is the learning rate. Repeating the above procedure for many batches yields the suitable parameter ω, which leads to the Bayesian-optimized force.

Code Availability

Data Availability

References

1. Schultz, W. & Dickinson, A. Neuronal coding of prediction errors. *Annu. Rev. Neurosci.* **23**, 473–500 (2000).
2. Gold, J. I. & Shadlen, M. N. The neural basis of decision making. *Annu. Rev. Neurosci.* **30**, 535–574 (2007).
3. Chatham, C. H. & Badre, D. Multiple gates on working memory. *Curr. Opin. Behav. Sci.* **1**, 23–31 (2015).

612 4. Miller, E. K. & Cohen, J. D. An integrative theory of prefrontal cortex function. *Annu. Rev.*
613 *Neurosci.* **24**, 167–202 (2001).

614 5. Robinson, D. A. Integrating with neurons. *Annu. Rev. Neurosci.* **12**, 33–45 (1989).

615 6. Seung, H. S. How the brain keeps the eyes still. *Proc. Natl. Acad. Sci. U. S. A.* **93**, 13339–13344
616 (1996).

617 7. Chafee, M. V. & Goldman-Rakic, P. S. Inactivation of parietal and prefrontal cortex reveals
618 interdependence of neural activity during memory-guided saccades. *J. Neurophysiol.* **83**, 1550–
619 1566 (2000).

620 8. Constantinidis, C., Funahashi, S., Lee, D., Murray, J. D., Qi, X. L., Wang, M. & Arnsten, A. F. T.
621 Persistent spiking activity underlies working memory. *J. Neurosci.* **38**, 7020–7028 (2018).

622 9. Funahashi, S., Bruce, C. J. & Goldman-Rakic, P. S. Mnemonic coding of visual space in the
623 monkey's dorsolateral prefrontal cortex. *J. Neurophysiol.* **61**, 331–349 (1989).

624 10. Fuster, J. M. & Alexander, G. E. Neuron activity related to short-term memory. *Science (80-.).*
625 **173**, 652–654 (1971).

626 11. Guo, Z. V., Inagaki, H. K., Daie, K., Druckmann, S., Gerfen, C. R. & Svoboda, K. Maintenance of
627 persistent activity in a frontal thalamocortical loop. *Nature* **545**, 181–186 (2017).

628 12. Kubota, K. & Niki, H. Prefrontal cortical unit activity and delayed alternation performance in
629 monkeys. *J. Neurophysiol.* **34**, 337–347 (1971).

630 13. Miller, E. K., Erickson, C. A. & Desimone, R. Neural mechanisms of visual working memory in
631 prefrontal cortex of the macaque. *J. Neurosci.* **16**, 5154–5167 (1996).

632 14. Murray, J. D., Bernacchia, A., Roy, N. A., Constantinidis, C., Romo, R. & Wang, X. J. Stable
633 population coding for working memory coexists with heterogeneous neural dynamics in
634 prefrontal cortex. *Proc. Natl. Acad. Sci. U. S. A.* **114**, 394–399 (2017).

635 15. Romo, R., Brody, C. D., Hernández, A. & Lemus, L. Neuronal correlates of parametric working
636 memory in the prefrontal cortex. *Nature* **399**, 470–473 (1999).

637 16. Baeg, E. H., Kim, Y. B., Huh, K., Mook-Jung, I., Kim, H. T. & Jung, M. W. Dynamics of population
638 code for working memory in the prefrontal cortex. *Neuron* **40**, 177–188 (2003).

17. Brody, C. D., Hernández, A., Zainos, A. & Romo, R. Timing and Neural Encoding of Somatosensory Parametric Working Memory in Macaque Prefrontal Cortex. *Cereb. Cortex* **13**, 1196–1207 (2003).

18. Cavanagh, S. E., Towers, J. P., Wallis, J. D., Hunt, L. T. & Kennerley, S. W. Reconciling persistent and dynamic hypotheses of working memory coding in prefrontal cortex. *Nat. Commun.* **9**, 1–16 (2018).

19. Chaisangmongkon, W., Swaminathan, S. K., Freedman, D. J. & Wang, X. J. Computing by Robust Transience: How the Fronto-Parietal Network Performs Sequential, Category-Based Decisions. *Neuron* **93**, 1504-1517.e4 (2017).

20. Harvey, C. D., Coen, P. & Tank, D. W. Choice-specific sequences in parietal cortex during a virtual-navigation decision task. *Nature* **484**, 62–68 (2012).

21. Spaak, E., Watanabe, K., Funahashi, S. & Stokes, M. G. Stable and dynamic coding for working memory in primate prefrontal cortex. *J. Neurosci.* **37**, 6503–6516 (2017).

22. Sreenivasan, K. K., Curtis, C. E. & D'Esposito, M. Revisiting the role of persistent neural activity during working memory. *Trends Cogn. Sci.* **18**, 82–89 (2014).

23. Wolff, M. J., Jochim, J., Akyürek, E. G. & Stokes, M. G. Dynamic hidden states underlying working-memory-guided behavior. *Nat. Neurosci.* **20**, 864–871 (2017).

24. Zhu, J., Cheng, Q., Chen, Y., Fan, H., Han, Z., Hou, R., Chen, Z. & Li, C. T. Transient Delay-Period Activity of Agranular Insular Cortex Controls Working Memory Maintenance in Learning Novel Tasks. *Neuron* **105**, 934-946.e5 (2020).

25. Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci. U. S. A.* **79**, 2554–2558 (1982).

26. O'Reilly, R. C., Mozer, M., Munakata, Y. & Miyake, A. Discrete Representations in Working Memory: A Hypothesis and Computational Investigations. *Second Int. Conf. Cogn. Sci.* 183–188 (1999).

27. Ferrell, J. E. Self-perpetuating states in signal transduction: Positive feedback, double-negative feedback and bistability. *Curr. Opin. Cell Biol.* **14**, 140–148 (2002).

28. Compte, A., Brunel, N., Goldman-Rakic, P. S. & Wang, X. J. Synaptic mechanisms and network dynamics underlying spatial working memory in a cortical network model. *Cereb. Cortex* **10**, 910–923 (2000).

29. Burak, Y. & Fieted, I. R. Fundamental limits on persistent activity in networks of noisy neurons. *Proc. Natl. Acad. Sci. U. S. A.* **109**, 17645–17650 (2012).

30. Kilpatrick, Z. P., Ermentrout, B. & Doiron, B. Optimizing working memory with heterogeneity of recurrent cortical excitation. *J. Neurosci.* **33**, 18999–19011 (2013).

31. Koulakov, A. A., Raghavachari, S., Kepecs, A. & Lisman, J. E. Model for a robust neural integrator. *Nat. Neurosci.* **5**, 775–782 (2002).

32. Brody, C. D., Romo, R. & Kepecs, A. Basic mechanisms for graded persistent activity: discrete attractors, continuous attractors, and dynamic representations. *Curr. Opin. Neurobiol.* **13**, 204–211 (2003).

33. Bouchacourt, F. & Buschman, T. J. A Flexible Model of Working Memory. *Neuron* **103**, 147-160.e8 (2019).

34. Druckmann, S. & Chklovskii, D. B. Neuronal circuits underlying persistent representations despite time varying activity. *Curr. Biol.* **22**, 2095–2103 (2012).

35. Bondanelli, G. & Ostojic, S. *Coding with transient trajectories in recurrent neural networks*. *PLoS Computational Biology* vol. 16 (2020).

36. Barak, O. & Tsodyks, M. Working models of working memory. *Curr. Opin. Neurobiol.* **25**, 20–24 (2014).

37. Ghazizadeh, E. & Ching, S. N. Slow manifolds within network dynamics encode working memory efficiently and robustly. *PLoS Comput. Biol.* **17**, 1–20 (2021).

38. Bi, Z. & Zhou, C. Understanding the computation of time using neural network models. *Proc. Natl. Acad. Sci. U. S. A.* **117**, 10530–10540 (2020).

39. Lundqvist, M., Herman, P. & Miller, E. K. Working memory: Delay activity, yes! persistent activity? maybe not. *J. Neurosci.* **38**, 7013–7019 (2018).

40. 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).

41. Constantinidis, C., Franowicz, M. N. & Goldman-Rakic, P. S. Coding specificity in cortical microcircuits: A multiple-electrode analysis of primate prefrontal cortex. *J. Neurosci.* **21**, 3646–3655 (2001).

696 42. Wimmer, K., Nykamp, D. Q., Constantinidis, C. & Compte, A. Bump attractor dynamics in
697 prefrontal cortex explains behavioral precision in spatial working memory. *Nat. Neurosci.* **17**,
698 431–439 (2014).

699 43. Fougnie, D., Suchow, J. W. & Alvarez, G. A. Variability in the quality of visual working memory.
700 *Nat. Commun.* **3**, 1229 (2012).

701 44. Pertzov, Y., Manohar, S. & Husain, M. Rapid forgetting results from competition over time
702 between items in visual working memory. *J. Exp. Psychol. Learn. Mem. Cogn.* **43**, 528–536 (2017).

703 45. Panichello, M. F., DePasquale, B., Pillow, J. W. & Buschman, T. J. Error-correcting dynamics in
704 visual working memory. *Nat. Commun.* **10**, 1–11 (2019).

705 46. Bae, G. Y., Olkkonen, M., Allred, S. R., Wilson, C. & Flombaum, J. I. Stimulus-specific variability in
706 color working memory with delayed estimation. *J. Vis.* **14**, 1–23 (2014).

707 47. Yendrikhovskij, S. N. Computing color categories from statistics of natural images. *J. IMAGING Sci.*
708 *Technol.* **45**, 409–417 (2001).

709 48. Papadimitriou, C., Ferdoash, A. & Snyder, L. H. Ghosts in the machine: Memory interference from
710 the previous trial. *J. Neurophysiol.* **113**, 567–577 (2015).

711 49. Kilpatrick, Z. P. Synaptic mechanisms of interference in working memory. *Sci. Rep.* **8**, 1–20 (2018).

712 50. Barbosa, J. & Compte, A. Build-up of serial dependence in color working memory. *Sci. Rep.* **10**, 1–
713 7 (2020).

714 51. Inagaki, H. K., Fontolan, L., Romani, S. & Svoboda, K. Discrete attractor dynamics underlies
715 persistent activity in the frontal cortex. *Nature* **566**, 212–217 (2019).

716 52. Russo, A. A., Bittner, S. R., Perkins, S. M., Seely, J. S., London, B. M., Lara, A. H., Miri, A., Marshall,
717 N. J., Kohn, A., Jessell, T. M., Abbott, L. F., Cunningham, J. P. & Churchland, M. M. Motor Cortex
718 Embeds Muscle-like Commands in an Untangled Population Response. *Neuron* **97**, 953-966.e8
719 (2018).

720 53. Balasubramaniam, R., Haegens, S., Jazayeri, M., Merchant, H. & Sternad, D. Neural Encoding and
721 Representation of Time for Sensorimotor Control and Learning. *J. Neurosci.* **41**, 866–872 (2021).

54. 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).

55. Barbosa, J., Stein, H., Martinez, R. L., Galan-Gadea, A., Li, S., Dalmau, J., Adam, K. C. S., Valls-Solé, J., Constantinidis, C. & Compte, A. Interplay between persistent activity and activity-silent dynamics in the prefrontal cortex underlies serial biases in working memory. *Nat. Neurosci.* **23**, 1016–1024 (2020).

56. Glaze, C. M., Kable, J. W. & Gold, J. I. Normative evidence accumulation in unpredictable environments. *Elife* **4**, 1–27 (2015).

57. Beck, J. M., Ma, W. J., Kiani, R., Hanks, T., Churchland, A. K., Roitman, J., Shadlen, M. N., Latham, P. E. & Pouget, A. Probabilistic Population Codes for Bayesian Decision Making. *Neuron* **60**, 1142–1152 (2008).

58. Sohn, H., Narain, D., Meirhaeghe, N. & Jazayeri, M. Bayesian Computation through Cortical Latent Dynamics. *Neuron* **103**, 934-947.e5 (2019).

59. Sohn, H. & Narain, D. Neural implementations of Bayesian inference. *Curr. Opin. Neurobiol.* **70**, 121–129 (2021).

60. Wang, J., Narain, D., Hosseini, E. A. & Jazayeri, M. Flexible timing by temporal scaling of cortical responses. *Nat. Neurosci.* **21**, 102–112 (2018).

61. Kingma, D. P. & Ba, J. L. Adam: A method for stochastic optimization. *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.* 1–15 (2015).

62. Sussillo, D. & Barak, O. Opening the black box: Low-dimensional dynamics in high-dimensional recurrent neural networks. *Neural Comput.* **25**, 626–649 (2013).

Acknowledgements

We thank R. Wessel and S. Ching for providing helpful comments to further improve the writing and presentation. This work was supported by Hong Kong Baptist University Strategic Development Fund, the Hong Kong Research Grant Council (GRF12200620), the Hong Kong Baptist University Research

Committee Interdisciplinary Research Clusters Matching Scheme 2018/19 (RC-IRCMs/18 19/SCI01) and the National Science Foundation of China (Grant 11975194) to C.Z..

Author contributions

Z.Y., L.T., and C.Z. designed research; Z.Y. performed research; Z.Y. analysed data; and Z.Y., L.T., and C.Z. wrote the paper

Ethics declarations

The authors declare no competing interests.

Supplementary Information