PNAS RESEARCH ARTICLE | NEUROSCIENCE
BIOPHYSICS AND COMPUTATIONAL BIOLOGY OPEN ACCESS

Chaotic neural dynamics facilitate probabilistic computations through sampling

Yu Terada[a,b,c,1] and Taro Toyoizumi[a,d]

Edited by Terrence Sejnowski, Salk Institute for Biological Studies, La Jolla, CA; received July 28, 2023; accepted February 13, 2024

Cortical neurons exhibit highly variable responses over trials and time. Theoretical works posit that this variability arises potentially from chaotic network dynamics of recurrently connected neurons. Here, we demonstrate that chaotic neural dynamics, formed through synaptic learning, allow networks to perform sensory cue integration in a sampling-based implementation. We show that the emergent chaotic dynamics provide neural substrates for generating samples not only of a static variable but also of a dynamical trajectory, where generic recurrent networks acquire these abilities with a biologically plausible learning rule through trial and error. Furthermore, the networks generalize their experience in the stimulus-evoked samples to the inference without partial or all sensory information, which suggests a computational role of spontaneous activity as a representation of the priors as well as a tractable biological computation for marginal distributions. These findings suggest that chaotic neural dynamics may serve for the brain function as a Bayesian generative model.

computational neuroscience | recurrent neural networks | chaos | Bayesian computation | cue integration

Significance

Neurons in the cortex show irregular activity with and without explicit stimuli. Theoretical modeling provides a persuading explanation of such complex dynamics using chaos—however, little is known about how chaotic neural dynamics contribute to neural computation. This work employs recurrent neural networks trained with biologically plausible learning to demonstrate that their chaotic dynamics enable the networks to perform the Bayesian computation through sampling. We consider cue-integration tasks requiring probabilistic computation, for which the recurrent neural networks can utilize their irregular dynamics to represent probabilistic distributions and generalize their experience to novel situations with partly missing inputs. The results suggest neural circuits can operate as generative models using inherent chaotic dynamics.

Humans and other animals face environments inherently associated with uncertainty, necessitating the handling and integration of uncertain information for survival. A wide spectrum of experimental studies has shown that the brain can perform nearly optimal Bayesian computation (1–6). While some computational models (2, 7) assume that neurons encode statistics of the underlying probability distribution, others suggest that neurons encode Monte Carlo samples drawn from the distributions (8–21). For the latter models, variability in neural activity is an essential element for probabilistic information representation.

Consistently, recent experiments have recorded a large number of neurons simultaneously and revealed that irregular patterns of neural activity (22, 23) underlie information processing in the brain. Such variability is generated spontaneously even in the absence of explicit changes in sensory input (24–26). At a macroscopic scale, functional MRI observations reveal specific patterns of intrinsic variability during the resting state. These patterns, known as default mode networks (27), exhibit structures that reflect experience and knowledge (28). At a microscopic scale, neural avalanches, in which neurons exhibit events with strong synchrony and burst-type activity with power-law distributions of sizes and lifetimes, are observed during spontaneous neural activity (29). Thus, irregular spontaneous neural activity ubiquitously emerges across various spatiotemporal scales.

The biological source of neural variability is currently under debate (30–32). Neural variability has sometimes been modeled by different types of stochastic noise in neural dynamics (33–35), such as input noises to neurons (14) or stochastic spiking due to spike-threshold noise in point-process-type models (36). Another factor of neural variability arises from vesicular transmission by synapses (37), which is modeled by vesicular release probability. These works assume the existence of a random number generator separately from the modeled neural circuits. In contrast to these stochastic models, our study focuses on models that explain neural variability in deterministic systems (38–45). Strong and heterogeneous synaptic connections with the overall balance between excitatory and inhibitory drives can endow even deterministic neural networks with the ability to generate high-dimensional variability by chaos. Some experimental observations support this hypothesis (31). This line of research has also triggered theoretical works elucidating the computational advantages of neural dynamics at the edge of chaos (46–49). Further, chaotic network states are shown to be a suitable initial condition that enhances learning efficiency (50–54). However, these neural networks do not typically exhibit chaos after learning, and their chaoticity has not been utilized explicitly for computation in trained networks. More recently, chaotic dynamics are shown to permit multiple time scales even close to marginally stable systems, which is advantageous for enabling long

Author contributions: Y.T. and T.T. designed research; Y.T. performed research; Y.T. analyzed data; and Y.T. and T.T. wrote the paper.

The authors declare no competing interest.

This article is a PNAS Direct Submission.

[1]To whom correspondence may be addressed. Email: yuterada@ucsd.edu.

This article contains supporting information online at https://www.pnas.org/lookup/suppl/doi:10.1073/pnas.2312992121/-/DCSupplemental.

Published April 22, 2024.

short-term memory without finely tuning parameters (44, 55). In this paper, we discriminate two types of neural variability arising from the different mechanisms, noise-driven and chaos-driven variability, and focus on the latter.

Previous Bayesian computation models of the brain (7–17) assume the existence of noise sources, which add stochastic variability. Hence, whether chaotic variability is compatible with biological Bayesian computation is unclear, especially because the chaotic property that expands the effect of perturbation in time is apparently contradictory to accurate computation. In addition, these stochastic models often rely on a parametric family to describe target probability distributions, which requires hand-crafted designs in advance. In biological setups, considering generic network architecture at the early stage of learning would be more plausible. While nonparametric methods to learn a target probability distribution are popular in machine learning (56–59), it is not trivial how the brain can implement these computations in a biologically plausible manner.

Here, we consider a nonparametric model to generate samples from the Bayesian posterior distribution. The sampling is achieved by utilizing chaotic network dynamics of recurrently connected deterministic neurons. During training, we use the biologically plausible node perturbation learning rule (60–63), which is represented as a "three-factor learning rule." Three-factor learning rules are promising candidates for enabling flexible adaptation for biological neural networks in a broad range of tasks (64–67) as they can be implemented with local computation, namely, by a Hebbian learning rule modulated by a global signal (68–70). We explore the efficiency of a three-factor learning rule.

We consider representative examples of Bayesian tasks to demonstrate that chaotic neural dynamics serve as the substrate for representing posterior probability distributions. Specifically, we use paradigmatic cognitive tasks for integrating information from multiple sources similar to human and animal tasks, known as cue-integration tasks (71, 72). We demonstrate that, after training using a local learning rule, chaotic recurrent neural networks learn to sample hidden static states or dynamic trajectories of a hidden variable in a near-optimal manner. We discuss the implications of using chaos for probabilistic computations in the brain.

Results

Neural Sampling through Chaotic Dynamics. We developed a rate-based recurrent network model that uses their intrinsic dynamics to draw samples of the hidden variable of interest from an estimated Bayes posterior distribution. These neurons are governed by discrete-time dynamics receiving inputs from their recurrent synapses and sensory neurons (see below and *Methods*). A simple cue-integration task is described in Fig. 1*A*. In this task, recurrent networks infer the probability distribution of the hidden variable θ of interest. For example, θ may describe the angular position of an object. For simplicity, we assume that the angular position can take one among n possible directions (a

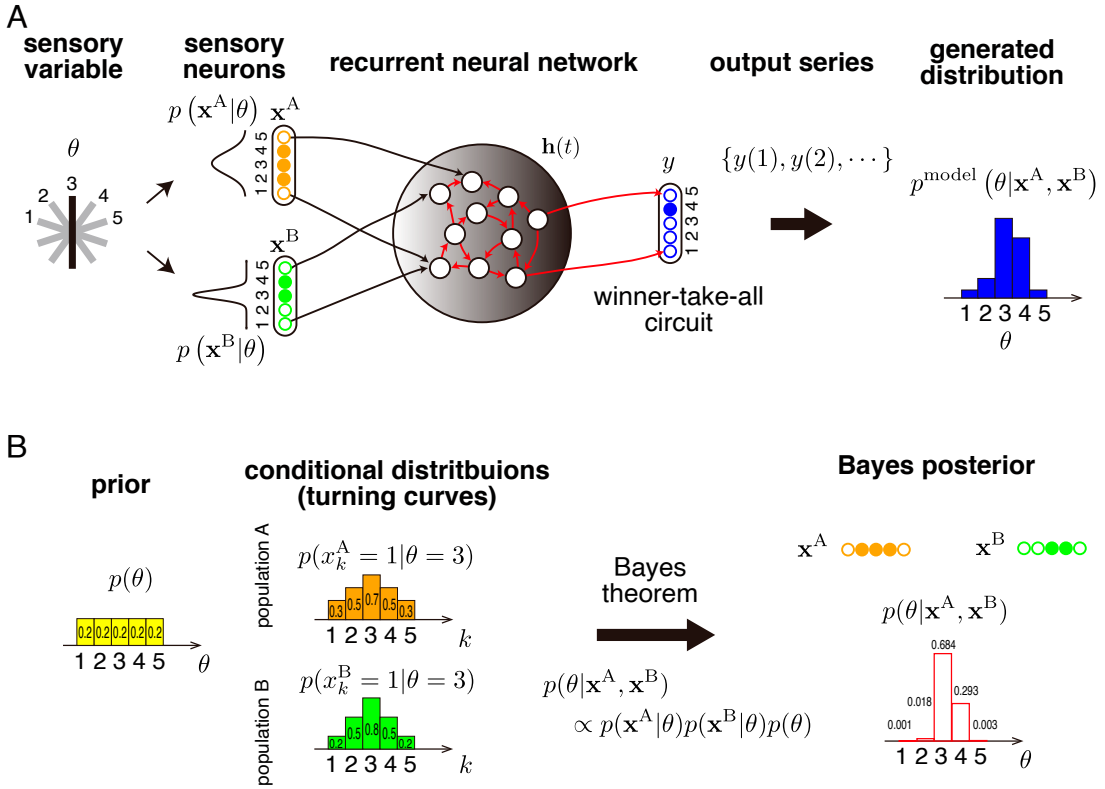

Fig. 1. (*A*) Schematics of the cue-integration task. The recurrent neural network receives inputs $\mathbf{x}^A$ and $\mathbf{x}^B$ as sensory inputs, whose firing patterns are determined stochastically based on their tuning curves and a hidden variable θ of interest. The network learns to infer the probability distribution of θ given these inputs and generates samples through the output activity. The network's output histogram composes an estimate of the posterior probability. During the learning period, the bias parameters as well as the internal and readout connections in red are modified. (*B*) The Bayes optimal posteriors in the cue-integration task. The prior of θ (the uniform distribution here) is shown with the yellow histogram on the *Left*. The conditional distributions of the two input vectors are specified by their tuning curve properties. The tuning curves describe the probability that each input neuron is active given θ; they are depicted using the orange and green histograms in the middle for $\theta = 3$. The Bayes posterior distribution given these input patterns $\mathbf{x}^A$, $\mathbf{x}^B$ is specified by the Bayes theorem and indicated by the red blank histogram on the *Right*.

biologically plausible extension will be discussed in *Discussion*). A recurrent neural network receives two kinds of input vectors $\mathbf{x}^\alpha$, which represent the firing patterns of two sets of sensory neurons $\alpha = A, B$. For example, the sets A and B may represent ensembles of auditory and visual sensory neurons, respectively, that encode the direction θ of the object. The lengths of these input vectors are also assumed n, and each element k takes either the active or inactive state, that is, $x_k^\alpha \in \{0, 1\}$. At the beginning of each trial, the value of θ is randomly drawn from a uniform distribution unless stated otherwise (nonuniform cases will be shown later), and the input elements are generated independently according to the conditional probability $p(x_k^\alpha|\theta)$ for an element $k = 1, 2, \ldots, n$ and a population α. This conditional probability specifies the tuning curve of neurons, characterizing the sensory neurons' activation probability given the angular position. It peaks at $k = \theta$ and decreases (slowly for $\alpha = A$ and rapidly for $\alpha = B$) as k deviates from the peak (an example is shown in Fig. 1*B*; see also *Methods*). Note that the network receives the information of θ only through noisy firing patterns $\mathbf{x}^\alpha$ of the sensory neurons. Within each trial of its length $T = 200$ time steps, the values of θ and $\mathbf{x}^\alpha$ are fixed in time. The discrete-time dynamics of the N recurrently connected neurons at time t ($t = 1, 2, \ldots, T$) are described by $\mathbf{h}(t) = J\boldsymbol{\phi}(\mathbf{h}(t-1)) + \sum_{\alpha=A,B} K^\alpha \mathbf{x}^\alpha + \mathbf{c}$, where $\mathbf{h}$ is a N-dimensional vector of recurrent neural network activity, J is a $N \times N$ matrix of recurrent synaptic weights, K_α ($\alpha = A, B$) are the $N \times n$ synaptic weight matrices from input α, $\mathbf{c}$ is a N-dimensional vector of baseline input parameters, and $\boldsymbol{\phi}(\cdot) = \tanh(\cdot)$ is an activation function (*Methods*). The entries of J are initially drawn identically and independently from the Gaussian distribution with zero mean and SD $g/\sqrt{N}$, parameterized by the gain parameter g. Note that both J and $\mathbf{c}$ change during learning. The entries of K_α are drawn from the standard normal distribution, and they are fixed during the entire simulation. Finally, an n-dimensional vector $\mathbf{y}(t)$ represents the binary (0 or 1) activity of output neurons at time t. These neurons receive inputs $\mathbf{z}(t) = W\boldsymbol{\phi}(\mathbf{h}(t)) + \mathbf{b}$, where W denotes $n \times N$ matrix of readout weights and $\mathbf{b}$ is an n-dimensional vector of bias parameters, and only the output neuron that receives the maximum input is active while the remaining are inactive, which could be implemented by the winner-take-all (WTA) mechanism (73); the active output neuron at time t represents a sample of the hidden variable, namely, $y_{\hat{\theta}}(t) = 1$ and $y_{\gamma \neq \hat{\theta}}(t) = 0$ with $\hat{\theta}(t) \equiv \mathrm{argmax}_k z_k(t)$ ($k = 1, 2, \ldots, n$). Our goal is to train the network so that the network-generated histogram of $\hat{\theta}$ approximates well the Bayesian posterior $p(\theta|\mathbf{x}^A, \mathbf{x}^B)$ given the current sensory input pattern.

We adopted a local learning rule based on the node perturbation method (60–63) to update synaptic weights and bias parameters, which is considered more biologically plausible than machine learning algorithms that require, for example, error backpropagation (see the details in *Methods*). The error function for learning is defined as the square of the Hellinger distance (74), which measures the distance between the histograms of active output neurons within the time window and the Bayesian posterior. We assumed that the Hellinger distance is evaluated by an external network and transferred to the modeled network during learning. According to the node perturbation learning scheme, we updated adjustable parameters J, W, $\mathbf{c}$ and $\mathbf{b}$, but set the baseline parameter $\mathbf{c}$ as zero through simulations for the static tasks. For example, the update of synapse J_{ij} is described by the product of the global signal and a Hebbian term that correlates presynaptic activities and postsynaptic perturbations Eq. **7**. The stochastic perturbations for computing the gradient are turned off after the training, and therefore the network dynamics become completely deterministic.

While a random network does not exhibit adequate sampling before training, the trained network utilizes its irregular dynamics to represent the posterior distribution (Fig. 2). The neural variability in the recurrent network plays a crucial role in expressing the uncertainty of the hidden variable because the goal here is not only to estimate the most likely value of θ but also to generate samples from the posterior distribution. Moreover, the trained network succeeded in integrating uncertain information from two different sources and produced a good approximated posterior probability.

In Fig. 3*A*, the time course of the error (the squared Hellinger distance between the generated output histogram and the Bayesian posterior) is shown for 10 randomly initialized networks during learning. The task performance depends on which sensory inputs are available. The performance using input B alone is better compared to that using input A alone. This is because the neurons in B have a narrower tuning curve, and the input B is hence more informative than input A. The best performance is achieved when both inputs are available, highlighting the ability of the network to integrate multiple cues. The resulting distributions $p(\theta|\mathbf{x}^\mathrm{A}, \mathbf{x}^\mathrm{B})$ represented by the trained network for different patterns of sensory inputs, closely approximate the Bayesian posteriors over a variety of input vectors (Fig. 3*B*). As expected, the output distribution approaches the Bayesian posterior as the sampling duration T increases (*SI Appendix*, Fig. S1). Further, the adaptation of the synaptic connectivity J within the recurrent network reduces significantly the sensitivity of learning outcomes to the initial network configuration, compared to the so-called "reservoir scheme" that adjusts only the readout parameters W and $\mathbf{b}$ (see, initialization at the edge of chaos in *SI Appendix*, Fig. S2 and at the chaotic regime in *SI Appendix*, Fig. S3).

Randomly connected recurrent neural networks with strong synaptic interactions can exhibit chaotic dynamics. To investigate the nature of network dynamics and how they are related to task performance, we monitored the largest Lyapunov exponent as a measure of chaoticity and compared it with the error function during the learning period (Fig. 3*C*). The network is initially placed slightly below the edge of chaos with the synaptic gain parameter $g = 1.05$. Note that the dynamics are nonchaotic at $g = 1.05$ because the sensory inputs suppress chaos in random neural networks as demonstrated previously (e.g., ref. 75). The result shows that, as task performance improves through learning, the largest Lyapunov exponent increases from the negative to the positive side. It indicates that chaos emerges through learning while the network develops the capacity for probabilistic computation. This result is in contrast to the findings of previous works that trained nonchaotic dynamics (50–52, 54), and it highlights the utility of chaotic dynamics in probabilistic computation. The enhancement of chaoticity by learning was reported in associative memory (76), but our study directly links the computational ability of networks to chaos through the lens of representation of probabilistic distributions. While the training develops the chaotic dynamics from the edge of chaos, we found that starting from generic chaos could perform better in our setup, as shown in *SI Appendix*, Fig. S4*A*.

Chaotic neural dynamics render the networks to be highly sensitive to a perturbation, which might appear incompatible with accurate computation. Indeed, when the initial activity of neurons $\mathbf{h}(0)$ is slightly perturbed in the trained network, its

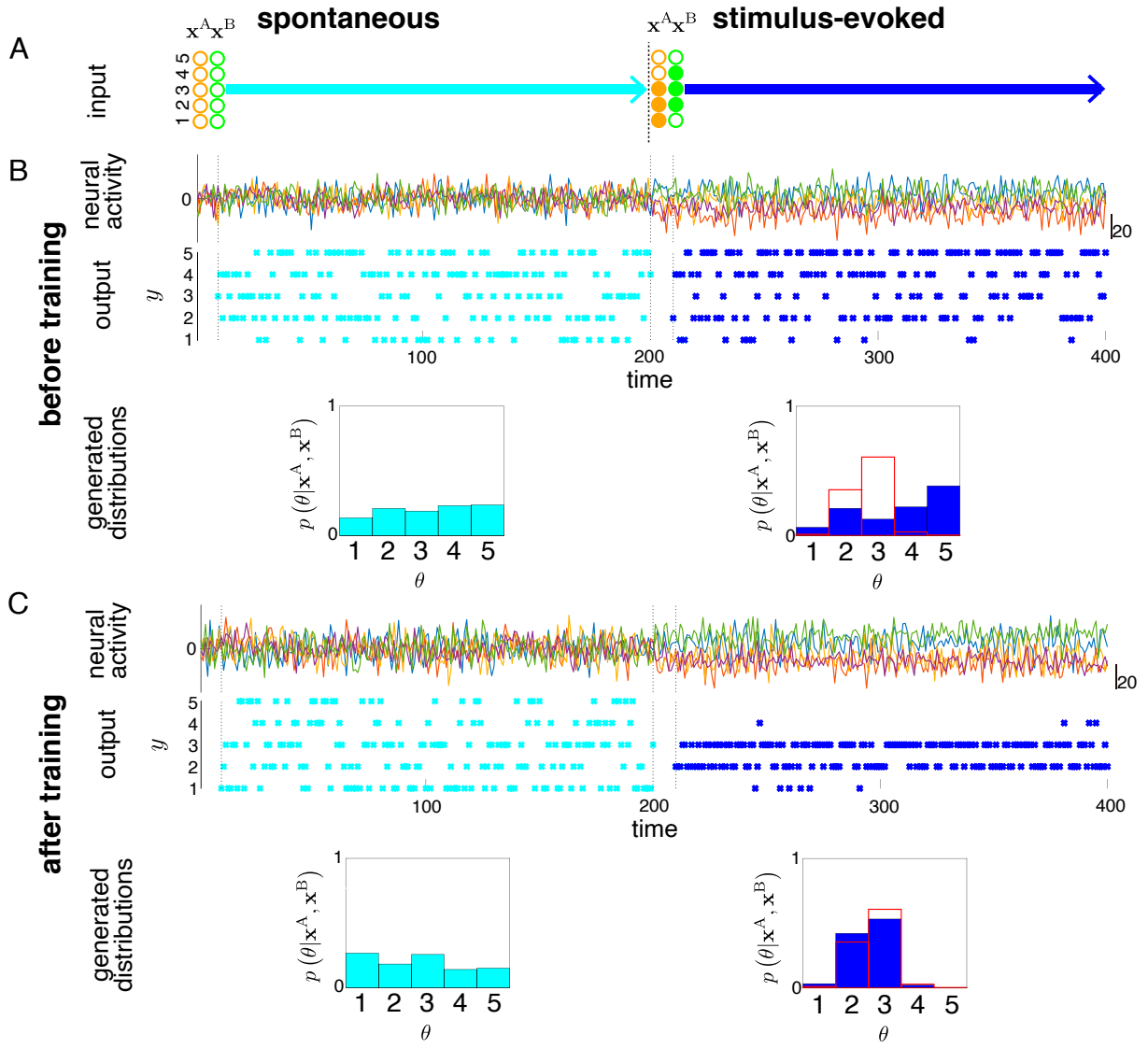

Fig. 2. An example of neural sampling before (with random initial weights) and after training. (*A*) The input patterns of the two sensory neuron populations. During the first-half period, sensory inputs are absent (all the sensory neurons are inactive). During the second-half period, the networks receive specific patterns of sensory inputs according to a hidden object direction. (*B* and *C*) (*Top*) The activities of a randomly selected subset of neurons, (*Middle*) the activities of output neurons (blue dots indicate active neurons), and (*Bottom*) the resulting histograms of active output neurons (colored bars). The generated distributions for the second half with blue-filled boxes are compared with the Bayesian posterior distributions, denoted by the red blank boxes.

subsequent trajectory changes drastically (Fig. 3*D*). Nonetheless, the output distributions in response to various sensory inputs are robust to the perturbation (see also *SI Appendix*, Fig. S5*A*). Hence, the high sensitivity to the initial condition does not compromise the reliability of the probabilistic computation. Furthermore, we demonstrate robust performance against perturbations in recurrent synaptic weights J (*SI Appendix*, Fig. S5*B*). We believe that this is a possible reason why the brain can perform reliable computation using highly variable neural dynamics.

Computational Role of Spontaneous and Evoked Neural Activities. As shown above, the recurrent networks draw output samples from distinct sets during spontaneous and evoked activities. The spontaneous activity of neurons may reflect the prior distribution over possible sensory inputs (5, 9). Here, we use our model to test the hypothesis that the learning process through stimulus-evoked samples can construct priors. In Fig. 2*C*, we assumed the uniform prior distribution over θ and obtained a near uniform output distribution during the spontaneous activity. However, the output distribution during spontaneous activity looked similar to the uniform prior even before the learning (Fig. 2*B*). Moreover, the learning process explicitly supervised the prior distribution by giving the networks training samples, including input sets without sensory stimulus. To address the problem of whether the network learns to code the prior distribution $p(\theta)$ during spontaneous activity from the learning only with stimulus-evoked samples, we trained networks using two different prior distributions (Fig. 4 *A* and *B*). The results show that the histogram of output during spontaneous activity matches the prior distribution $p(\theta)$ well. The learning curves for the errors for stimulus-evoked and spontaneous patterns indicate that the local learning sculpts chaotic dynamics not only for the posteriors but also for the prior only relying on stimulus-evoked samples (Fig. 4*C*).

We showed above that the network output is distributed approximately representing the prior distribution when both sensory inputs are absent. Note that the prior distribution is

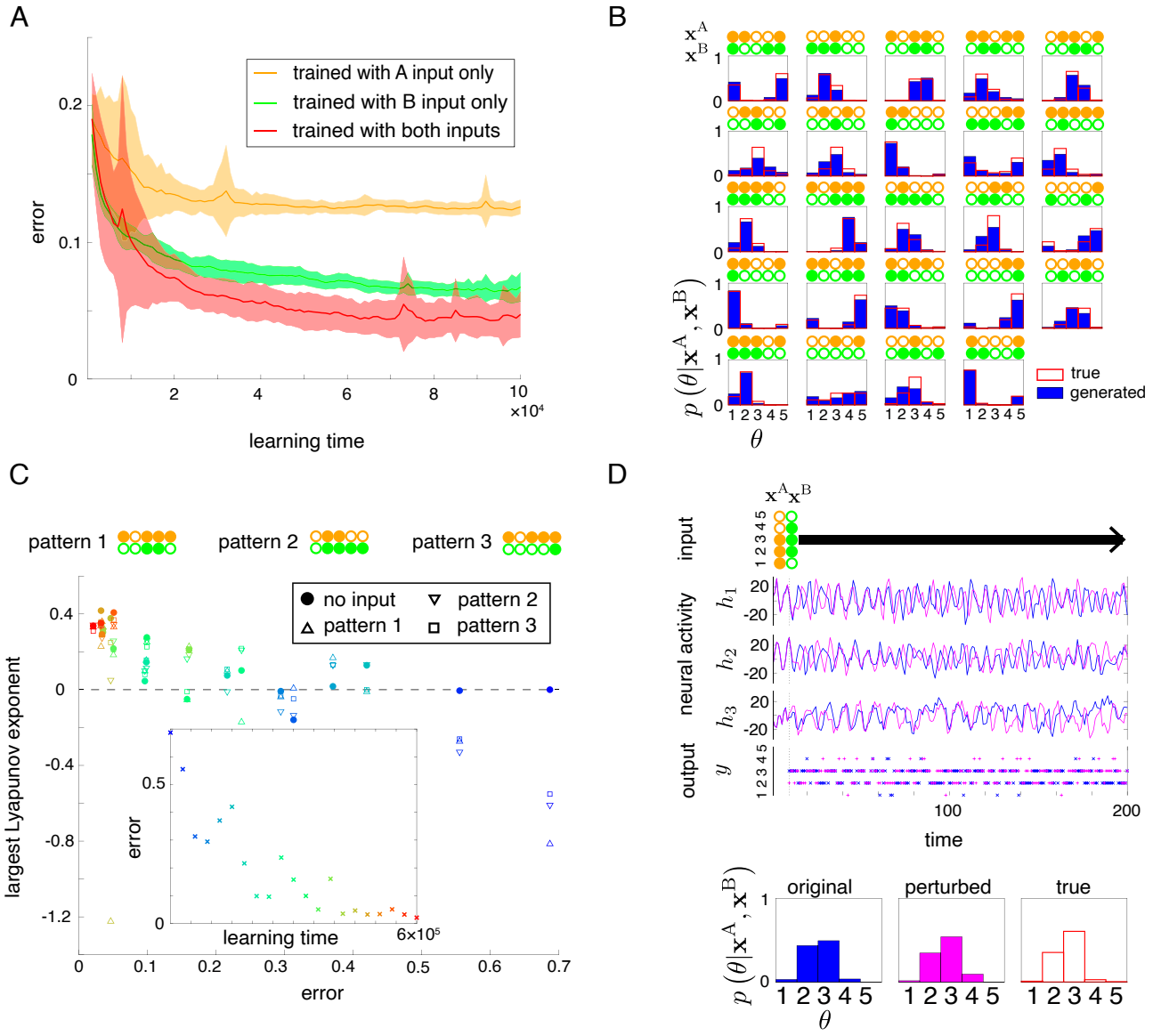

Fig. 3. Performance and chaoticity of the recurrent neural networks in the cue-integration task. (*A*) Learning curves of the networks that receive inputs from either of the populations (A or B) and from both during training, where the errors are evaluated for the posteriors given A and B. (*B*) Probability distributions provided by the trained network (the blue filled boxes) and true posteriors corresponding to the sensory inputs (red blank boxes) for multiple examples of input patterns. Input patterns A and B are shown above the distributions. (*C*) Largest Lyapunov exponent versus error value for networks during training for different patterns of inputs. The inset shows the errors versus learning time. The markers are colored from blue to red depending on the elapsed time. (*D*) *Top*: Input, dynamics of selected neurons, and output in the trained networks with two similar initial conditions, where the initial state of the magenta trajectory is slightly perturbed from that of the blue trajectory. *Bottom*: Estimated posterior distributions generated by the two networks and the true posterior.

written by averaging the posterior distribution with respect to the two unobserved sensory inputs, $p(\theta) = \mathbb{E}_{\mathbf{x}^A,\mathbf{x}^B}[p(\theta|\mathbf{x}^A,\mathbf{x}^B)]$, where $\mathbb{E}_{\mathbf{x}^A,\mathbf{x}^B}$ represents the expectation over $\mathbf{x}^A$ and $\mathbf{x}^B$. A natural extension of the above result is the case, where only one of the sensory inputs is observed. For example, if only $\mathbf{x}^A$ is observed, the output distribution might follow the conditional average of the posterior of the form $p(\theta|\mathbf{x}^A) = \mathbb{E}_{\mathbf{x}^B|\mathbf{x}^A}[p(\theta|\mathbf{x}^A,\mathbf{x}^B)]$, where $\mathbb{E}_{\mathbf{x}^B|\mathbf{x}^A}$ is the conditional average over $\mathbf{x}^B$ given $\mathbf{x}^A$. Our simulations demonstrate that this property holds with high accuracy. Fig. 5 shows the result in two cases where the trained networks receive inputs either from population A or B. We note that, as in Fig. 4, the networks were trained using only stimulus-evoked samples. Namely, the network receives nonzero inputs from at least one active neuron in both sensory populations. This situation can correspond to a condition in which an animal receives partial information (for example, only auditory or visual inputs) and generalizes its experience to this novel situation with more uncertainty. Conditionally averaging the posterior distribution in a direct manner can be computationally expensive because it involves the sum over potential unobserved input candidates, whose complexity grows exponentially with the number of unobserved variables. Moreover, the computation must be revised when the observed input changes. Through chaotic sampling, the networks succeed in approximating the conditionally averaged posteriors circumventing the evaluation of the sum over unobserved input candidates. This implementation constitutes a biologically suitable strategy to implement probabilistic computation with limited computational resources.

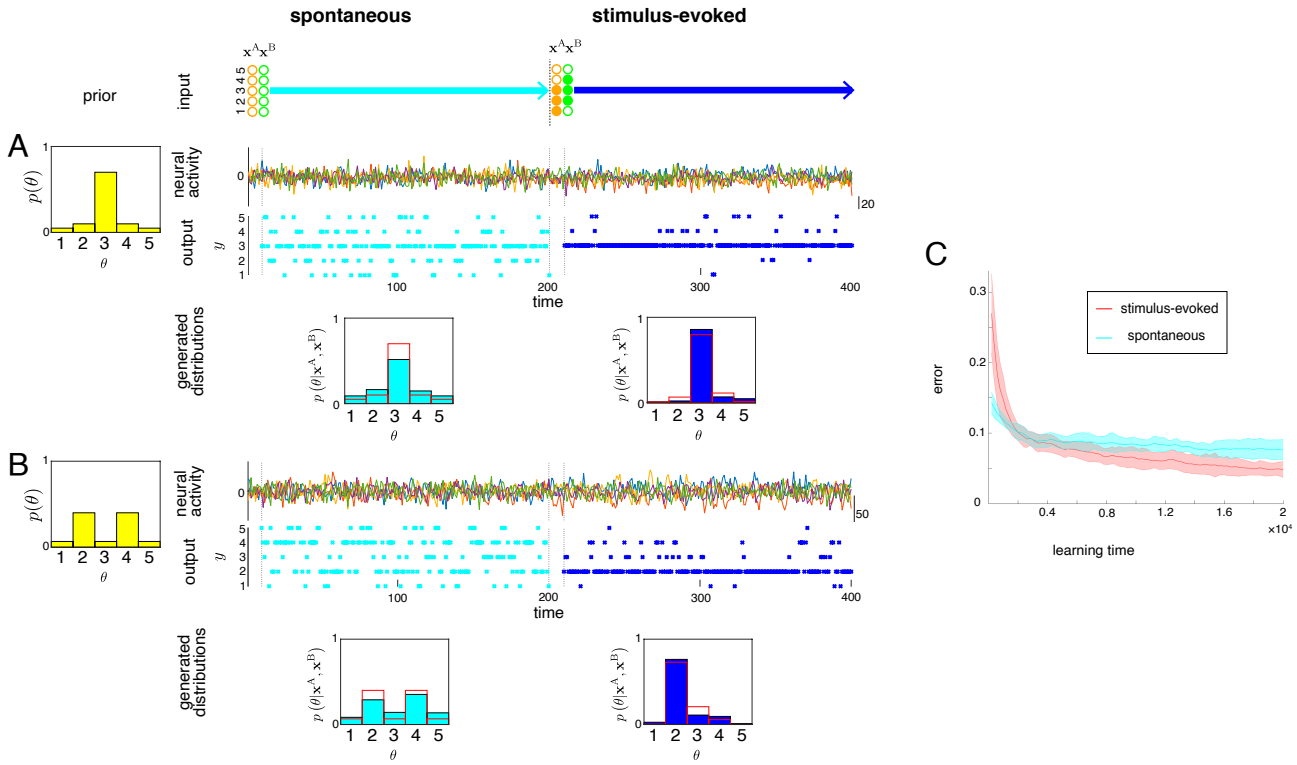

Fig. 4. (*A* and *B*) Neural sampling after learning with different priors. Sensory inputs are absent during the first-half period while a specific pattern of sensory inputs is injected during the second-half period. Two example priors, for *A* and *B*, are represented by the yellow histograms on the *Left*, respectively, which are distinct from the uniform distribution used in Fig. 2. The generated distributions during the two periods are shown below the neural activity traces and the outputs on the *Right*. The conventions are as in Fig. 2. (*C*) Learning curves for training errors with stimulus-evoked cases (red curve) and generalization errors where sensory inputs are absent (cyan curve). The true prior is the same as *A*.

Dynamical Probabilistic Inference. Next, we illustrate another example of probabilistic tasks that recurrent neural networks can implement using their chaotic dynamics. In the previous task, the networks exhibit their ability to integrate multimodal information to infer underlying static probability distributions. We hypothesize that the recurrent network model can also

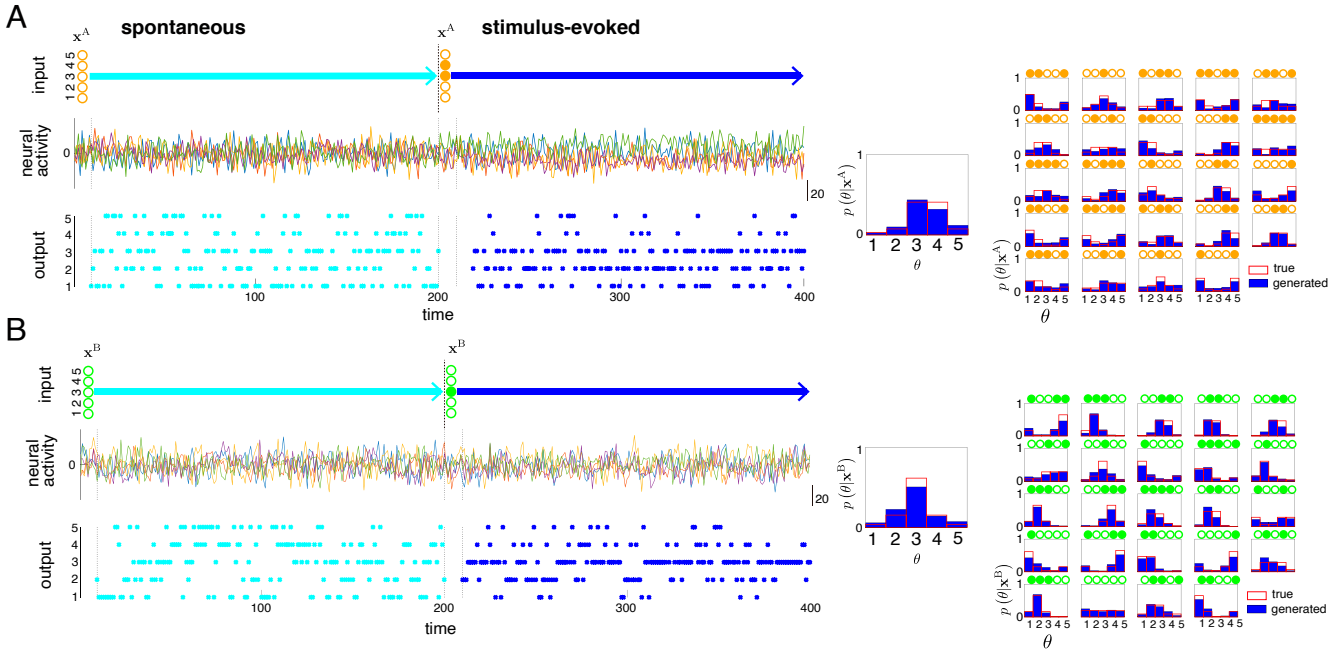

Fig. 5. Inference under a single cue by the network that was trained with the same procedure as Figs. 1–4. *Left*: Neural sampling for two cases given a single cue from A or B neurons in the trained networks. Sensory inputs are absent during the first-half period, and the networks receive the inputs only from one of the two populations (*A*: A population, *B*: B population) of sensory neurons during the second-half period. The conventions are as in Fig. 2. *Middle*: Generated distributions and Bayes posteriors for the case in the *Left* panel. *Right*: A set of examples that correspond to Fig. 3*B* but with only one of the sets of sensory neurons observed.

learn to generate samples of dynamic trajectories in a sensory-input-dependent manner. The task we consider here is one in which the network discriminates types of inputs and, depending on them, switches the transition probabilities of the outputs (schematics is shown in Fig. 6*A*). It requires the network to generate samples depending on its current internal state. To implement this computation, recurrent neural networks are required to learn temporal statistics. Here, we consider a simple setup, in which inputs A and B are both single-bit binary inputs, and we assume $n = 3$ states of θ and the corresponding output. The target transition probability matrices depend on sensory inputs; they have a uniform structure without sensory inputs and heterogeneous structures with sensory inputs (*Methods*). The same training scheme for the internal and readout parameters is applied again. We note that a network requires both sources of information as sensory inputs for accurate inference and, hence, this task is also a cue-integration task.

The trained recurrent networks integrate the information of the two inputs and utilize their chaotic variability to generate a dynamic output trajectory. Their learning curves are shown in Fig. 6*B*. The generated output trajectories approximately follow the target transition probabilities for both types of input patterns (Fig. 6*C*), while the time-averaged distributions of the outputs themselves are uninformative and almost uniform. Thus, the output dynamics driven by chaos can accurately reproduce the sequence of the target variable that follows the input-dependent stochastic dynamics. Hence, these networks perform a general form of dynamic sampling tasks based on Bayesian cue

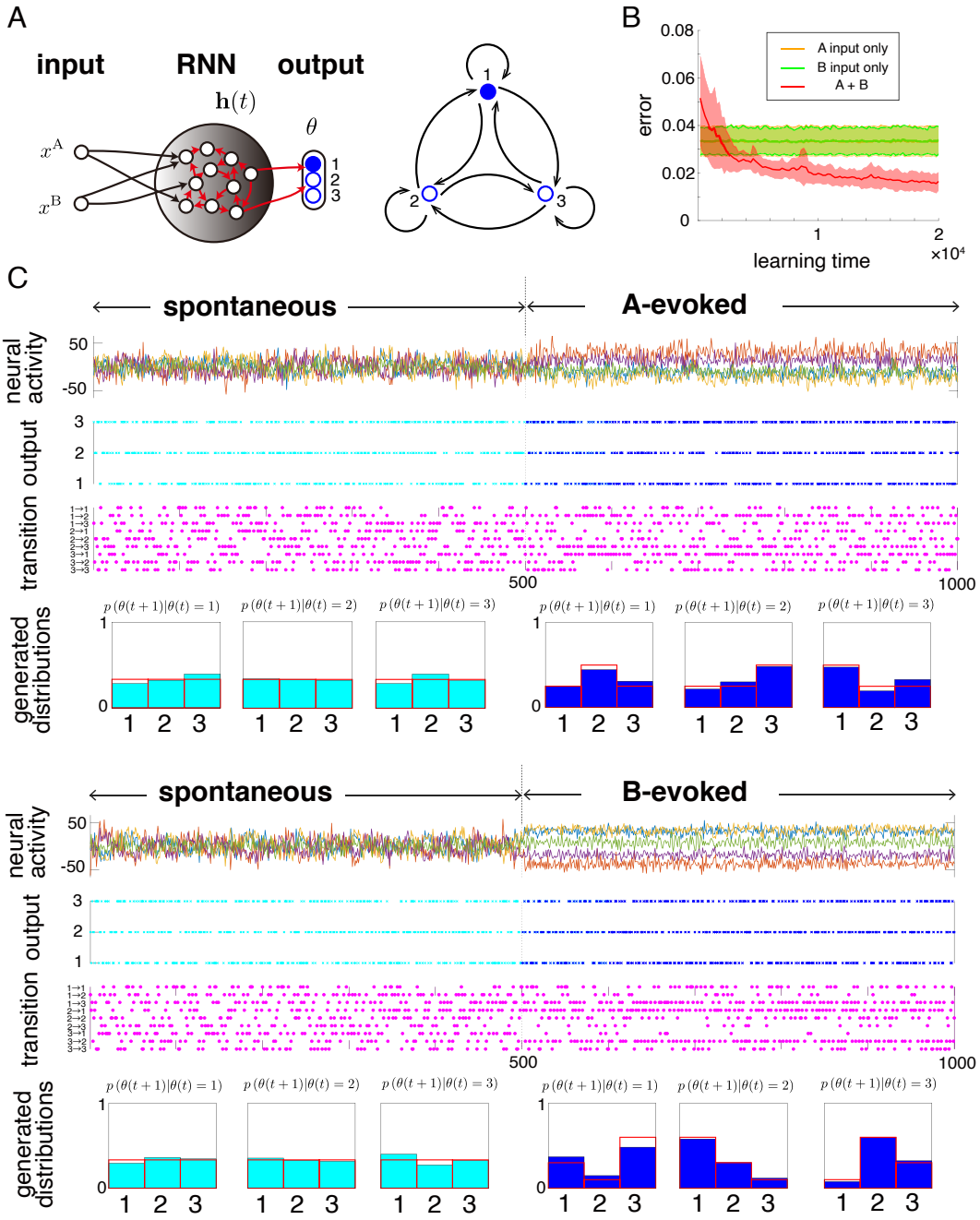

Fig. 6. Generating sample sequences with temporal relations depending on sensory inputs. (*A*) Schematics of the setting. (*Left*) Network setup. (*Right*) Markovian transitions between the output states. (*B*) Learning curves using either input or both. (*C*) Neural dynamics, output, transition types, and transition histograms after learning. The network receives no input in the first half and receives the sensory input $(x^A, x^B) = (1, 0)$ (*Top*) or $(0, 1)$ (*Bottom*) in the second half.

integration. While recurrent networks with external stochastic noise (for example, ref. 14) could achieve this computation, the stochastic implementation requires different computational components separately; the probability representation through stochastic inputs and state-dependency of computations via recurrent connections. By contrast, chaotic dynamics unify the two properties through recurrent network interactions in our model.

Discussion. We demonstrated that the recurrent neural networks trained using the local learning rule can integrate multimodal sensory inputs, compose near Bayes optimal posterior distributions, and draw samples from them by utilizing internally generated chaotic variability. Chaotic dynamics is apparently incompatible with reliable computation because of the butterfly effect that exponentially spreads dynamical trajectories initially started from nearby network states. Therefore, previous works have mostly avoided using a directly chaotic phase in the final state. However, as shown in Fig. 3*B* and *SI Appendix*, Fig. S5, this work offers a basis for realizing reliable probabilistic computation using chaotic neural dynamics. We speculate that this mechanism may underlie flexible information processing in the brain accompanying large variability, with the fact that experimental supports for chaotic neural dynamics are already reported (31). While networks with fixed-point dynamics can solve tasks not needing sampling (*SI Appendix*, Fig. S7), neural variability is widely observed in the brain, and tasks that require sampling, similar to ones considered in the main text, are relevant to animal behavior (77). Further evidence during Bayesian computation may become available in the future using advanced experimental techniques such as wide-field imaging (78, 79), dense electrophysiology recordings (80), and neural control by optogenetics (81); our results (Fig. 3*D* and *SI Appendix*, Fig. S5) imply that injection of a perturbation to some neurons can drastically change the network dynamics and resulting time-course of outputs due to chaotic network dynamics, but without significantly altering the inferred posterior distribution in the form of sampling histogram from which behaviors are generated. This prediction could be tested by confirming unaltered task performance despite the sensitivity of network states to perturbations during a statistical task. In contrast, such sensitivity would not be observed according to the noise-based sampling approach (9–12, 14, 82). Specifically, in order to test whether the distinction matters, we could compare two systems with deterministic and stochastic noises that generate apparently similar dynamics, by adjusting input statistics in the stochastic system (83). Their time correlations could be identical, but stochastic systems do not show a high sensitivity to a perturbation. This strategy may be useful to confront our results with experimental data.

We employed the node perturbation learning rule in the main text to train the recurrent networks in a trial-and-error way, where the network receives the scalar value of error information. To study the effect of learning rules, we also used random feedback local learning (67) as another biologically plausible learning rule as well as backpropagation (*SI Appendix*, Fig. S6). These learning rules require explicitly calculating the gradients. In this sense, we assume that the node perturbation learning rule may be simpler to implement in the brain. In addition, unlike the above gradient-based learning rules, the results suggest that the node perturbation learning rule can more robustly develop chaos even starting from a nonchaotic regime, consistently with a previous observation (63). While the gradient-based learning rules can cause the gradient vanishing problem to perform sampling at the network's fixed point, the node perturbation can explore a wider search range with appropriate perturbation amplitude.

One limitation of this work is that it does not address how the brain compares two probability distributions to compute the error values. The computation of error values could be learned through an unsupervised learning framework such as generative adversarial learning (58) by a biologically plausible learning rule like node perturbation or random feedback local learning. Future computational and experimental research is needed to study the biological mechanism underlying the evaluation of probabilistic errors.

We note that the use of chaotic dynamics in recurrent neural networks as a source of random-number generators has been investigated (84, 85). One work (84) proposes an engineering method, using a hand-crafted network architecture, to solve a static problem without addressing Bayesian cue integration. Another work (85) uses chaotic input generated in a separate network instead of assuming external stochastic input to solve static tasks. Hence, similar to standard generative models, these works assumed networks that learn to process input variability as opposed to generate variability, which seems a less flexible architecture because of separated networks for the representation and random-number generation. Neither of the above works addresses how biological systems could implement such computations, how networks with a generic initial structure can learn the task, or how spontaneous variability is related to evoked variability in the presence of an input. In contrast, we have shown not only that chaotic neural dynamics are useful for stochastic sampling in both static and dynamic tasks but also that generic networks can learn Bayesian computation and sampling through the biologically plausible synaptic plasticity rules. Importantly, as shown in Fig. 4, our model suggests that the ensemble of evoked activity during training shapes spontaneous activity, even when the training data do not include the corresponding samples for spontaneous activity. Further, the proposed networks not only express the prior distribution but also approximate conditionally averaged posteriors, such as $p(\theta|\mathbf{x}^{\mathrm{A}}) = \mathbb{E}_{\mathbf{x}^{\mathrm{B}}|\mathbf{x}^{\mathrm{A}}} p(\theta|\mathbf{x}^{\mathrm{A}}, \mathbf{x}^{\mathrm{B}})$, as its generalization. Our results in Fig. 5 showed that the network could represent this distribution for a variety of observed input $\mathbf{x}^{\mathrm{A}}$, simply by omitting input $\mathbf{x}^{\mathrm{B}}$ without needing to explicitly evaluate the computationally intense conditional average. This is an attractive property for brain-inspired computing.

In this work, we suggested the role of chaotic variability in sampling-based computations. However, it might also serve other roles. For example, during training, we applied the node perturbation algorithm that randomly perturbs neural activity and correlates the perturbation with future rewards. While the use of white-noise perturbations is suitable for uniformly exploring the activity space without assumptions, structured chaotic variability, shaped by experience, may be suitable for targeted exploration in the activity space (86). The use of internally generated chaotic variability to guide the exploration of node perturbation learning and comparison of its performance with the one using white noise is a potential direction for future research. Such structured neural variability might control the trade-off between exploration and exploitation in learning.

Notably, while we considered deterministic neural networks without noise for simplicity, chaotic dynamics are well defined in the presence of noise (75, 87). We do not rule out a contribution from noise in neural variability; however, we simply focused on the network mechanism that deterministically generates irregular neural dynamics via strong synapses, which could enhance

stochastic seeds of variability if any. We believe that future studies will identify their separable contributions and potential synergy. For example, additive noise can control the chaotic transition to maintain high memory capacity (87) and quenched noise can also modulate high-performance regime introducing multistability that includes chaotic dynamics (88).

Several extensions of the proposed model are possible. First, recurrent neural network models have offered interpretable and parsimonious accounts to bridge different types of cognitive behaviors and neural circuit activity (6, 89–92). Therefore, the Bayesian mechanism proposed with the simple and abstract recurrent neural network model could be tailored to explain more specific behaviors, such as motor control, psychophysics effects, and decision-making (35, 72). Second, we focused on discrete-time dynamics of neural networks for simplicity; however, extensions toward more biologically plausible models are also possible. For example, continuous-time models share many dynamical properties with our model and may become essential if more precise temporal patterns are needed. Moreover, although we set the time-scale parameter as $\tau = 1$, this assumption can be easily extended. Such studies could help for a comparison of time scale in recurrent neural networks with experimental observations. Third, we have assumed that neurons communicate with each other through their firing rates, but communication through spiking activity may play an important role. Importantly, the node perturbation learning rule that we used is directly applicable in spiking neural networks as well (62). It would be interesting to explore whether this difference may contribute to biological correspondence as well as energy and computational efficiency. Fourth, humans and other animals exhibit behavioral variability in their environments. Deterministic models explain the hierarchy of multiple timescales and the interplay among them (93, 94). Chaotic dynamics may help bridge neural and behavioral variability in the context of probabilistic tasks. Our results suggest that macroscopic variability in animal behaviors may partly arise from probabilistic sampling utilizing microscopic neural activity. Furthermore, although we only considered five angular positions, higher resolution than $2\pi/5$ is realized in the head direction systems in animals (for example ref. 95), and it does not seem appropriate to scale our computation with a larger readout network using the argmax function. One possible way to address this issue is to increase the number of output neurons but let them interact through fixed lateral couplings to achieve a biological tuning curve (96, 97). The estimated position may be read out as the center of the activity bump. Finally, we assumed that the tasks are ideally simplified and highly low-dimensional, which does not essentially require high-dimensional complex neural dynamics. To perform more complex tasks, the networks would need higher dimensionality, whose entropy exceeds that of target distributions. However, because previous work supports the low-dimensionality in neural dynamics (98), animals might sometimes approximate a complex distribution using low-dimensional neural dynamics for biological constraints or fast learning. In generic networks, high-dimensional chaotic dynamics are associated with strong chaos (45), which may sacrifice the performance as shown in *SI Appendix*, Figs. S4*A* and S6 *A* and *B*.

Methods

Model. A recurrent neural network model, where the N neurons are recurrently connected through the synaptic weights denoted as the matrix J, and the membrane potential variable of a neuron i at time t is represented as $h_i(t)$, was considered. The continuous version of the network models is defined for infinitesimal Δ as

$$\tau \frac{\mathbf{h}(t+\Delta)-\mathbf{h}(t)}{\Delta} = -\mathbf{h}(t) + J\boldsymbol{\phi}\left(\mathbf{h}(t)\right) + \sum_{\alpha=A,B} K^{\alpha}\mathbf{x}^{\alpha}(t) + \mathbf{c}, \quad [1]$$

where τ is the time-scale parameter, c_i represents the time-independent baseline of the activity of neuron i and $x_i^{\alpha}(t)$ represents the binary activation state of a sensory neuron i of the input population α, and fixed input weights are K^{α} (Figs. 1*A* and 6*A*). The output of the network at time t is specified by $\hat{\theta}(t) = \mathrm{argmax}_{\theta} z_{\theta}(t)$ $(\theta = 1, 2, \ldots, n)$ with $\mathbf{z}(t) = W\boldsymbol{\phi}(\mathbf{h}(t)) + \mathbf{b}$, where a row of W, $\mathbf{W}_{\theta}$ is the readout weight for the output θ, and b_{θ} is the readout bias. The activity of the output neurons is thus given by $y_{\hat{\theta}}(t) = 1$ and $y_{k\neq\hat{\theta}}(t) = 0$. For simplicity, we focus on the discrete-time dynamics of Eq. **1** by setting $\tau = 1$ and $\Delta = \tau$ and omitting the decay term as in the previous works (48, 75, 99). Indeed, the discrete dynamics share several critical properties with the continuous version (39, 48, 75, 87, 99). We will discuss an extension to continuous-time cases in Discussion. Sensory input patterns $x_i^{\alpha}(t)$ are fixed within one time window for inference for one posterior distribution, although they can change when the inference is switched. We denote as x_i^{α} for simplicity. In this paper, we omit the self-interactions, the effective decay terms as $J_{ii} = 0$.

Learning Rule. We adopted a biologically plausible learning rule, the node perturbation rule (62, 100), which requires only local computation and at the same time approximately reproduces the outcome of the error backpropagation learning. We assume noisy inputs during learning to induce perturbative exploration of synaptic weights and eligibility traces (101). In this learning rule, the networks exploit the global signals, which can be regarded as "the third factors" (68, 69) and are linked to errors or rewards. Additionally we define the error function as the square of the Hellinger distance between the true posterior distributions and the empirical distributions generated by the network; $E = H^2 = \sum_{\theta}(\sqrt{p_{\theta}} - \sqrt{q_{\theta}})^2/2$, where p_{θ} is the Bayesian posterior and q_{θ} is the generated histogram. The error was defined as a function of the empirical distribution of network output, which we evaluate as a histogram over a fixed time interval.

During the training period, we perturbed the systems by adding small stochastic noises: the neural dynamics during training are described as

$$\mathbf{h}(t) = J\boldsymbol{\phi}\left(\mathbf{h}(t-1)\right) + \sum_{\alpha=A,B} K^{\alpha}\mathbf{x}^{\alpha} + \mathbf{c} + \boldsymbol{\xi}(t), \quad [2]$$

$$\mathbf{z}(t) = W\boldsymbol{\phi}(\mathbf{h}(t)) + \mathbf{b} + \boldsymbol{\eta}(t), \quad [3]$$

where $\boldsymbol{\xi}(t)$ and $\boldsymbol{\eta}(t)$ are independent noises.

As an example, we illustrate the learning of J_{ij} in detail, while other cases follow similarly. The error function can be expanded over the power of the input trajectory as

$$E[\xi] \simeq E^0 + \sum_s \sum_j \frac{\partial E[\xi]}{\partial \xi_j(s)} \xi_j(s), \quad [4]$$

where E^0 denotes the baseline error value without the perturbation input. By averaging the product of $\xi_i(t)$ with this equation over the statistics of ξ with the Dirac delta correlation, we see

$$\left(E[\xi] - E^0\right)\xi_i(t) \propto \frac{\partial E[\xi]}{\partial \xi_i(t)} = \frac{\partial E[\xi]}{\partial h_i(t)} \quad [5]$$

holds. Using this relation, we obtain the expression for the gradient of the error function over J_{ij}:

$$\frac{\partial E}{\partial J_{ij}} = \sum_s \frac{\partial E}{\partial h_i(s)} \frac{\partial h_i(s)}{\partial J_{ij}} \propto \sum_s \left(E[\xi] - E^0\right)\xi_i(s)\boldsymbol{\phi}(h_j(s-1)). \quad [6]$$

We use the gradient descent method, and the resulting update equation for J_{ij} is written as

$$\Delta J_{ij} = -\epsilon_J \left(E^{\xi} - E^0\right) \sum_s \xi_i(s) \phi\left(h_j(s-1)\right), \quad [7]$$

where E^{ξ} and E^0 represent the error values under the existence and absence of the perturbation, respectively. Here, we take the sum over time-step s during a batch period. Similarly, for the other adaptive parameters

$$\Delta c_i = -\epsilon_h \left(E^{\xi} - E^0\right) \sum_s \xi_i(s), \quad [8]$$

$$\Delta W_{ij} = -\epsilon_W \left(E^{\xi} - E^0\right) \sum_s \eta_i(s) \phi\left(h_j(s)\right), \quad [9]$$

$$\Delta b_i = -\epsilon_b \left(E^{\xi} - E^0\right) \sum_s \eta_i(s). \quad [10]$$

Here, ϵ denotes the learning rate for each factor. We adopted the Adam method (102) to determine the learning rate, in which the hyperparameters are fixed as usually: $\epsilon^{\mathrm{Adam}} = 0.001, \rho_1^{\mathrm{Adam}} = 0.9, \rho_2^{\mathrm{Adam}} = 0.999$, and $\delta^{\mathrm{Adam}} = 10^{-8}$ for all parameters J, $\mathbf{c}$, W, and $\mathbf{b}$, while other schemes could also be used (52). Stochastic noises $\xi_i(t)$ and $\eta_i(t)$ are drawn from the uniform distributions.We draw them from the supports $[-1, 1]$ for $g = 8.0$ and $[-2, 2]$ for $g = 1.05$, respectively, if not stated otherwise. The effect of the noise amplitude is studied in *SI Appendix*, Fig. S4*B*. After the training period, the stochastic noises were no longer applied.

Static Cue-Integration Task. To study the capability of the chaotic neural networks in probabilistic computation, we considered a cue-integration task where the networks infer the posterior distributions from A and B inputs, as shown in Figs. 1–3. The networks received inputs from sensory neurons in populations A and B, whose binary activities (0 or 1) are determined stochastically according to the tuning curves described below.

We considered two different sensory populations A and B, each consisting of $n = 5$ neurons, and the recurrent neural network consisting of $N = 100$ neurons. We set different tuning curves for A and B (Fig. 1*B*): a neuron i in population A becomes active with the probability 0.7 for $\theta = i$, 0.5 for $\theta = i \pm 1 \pmod 5$, and 0.3 for $\theta = i \pm 2 \pmod 5$, respectively. For example, given the orientation value $\theta = 1$, we have the conditional probability of a neuron i firing $p(x_1^{\mathrm{a}} = 1|\theta = 1) = 0.7, p(x_2^{\mathrm{a}} = 1|\theta = 1) = p(x_5^{\mathrm{a}} = 1|\theta = 1) = 0.5$, $p(x_3^{\mathrm{a}} = 1|\theta = 1) = p(x_4^{\mathrm{a}} = 1|\theta = 1) = 0.3$. The tuning curve of a neuron i in B is specified with the probability 0.8 for $\theta = i$, 0.5 for $\theta = i \pm 1 \pmod 5$, and 0.2 for $\theta = i \pm 2 \pmod 5$, respectively. In Figs. 4 and 5, we removed the samples where all neurons in A or B are inactive; if $\mathbf{x}^{\mathrm{A/B}} = 0$ is produced, we draw the $\mathbf{x}^{\mathrm{A/B}}$ again. As a result, at least one neuron in A and B would be active and the networks thus receive nonzero stimuli from both populations.

We allocated a transient period of 10 time steps and a generative period of 190 time steps for each input.

In this task, the aim of a network is to generate probabilistic distributions by sampling the output y, whose distribution should be as close as possible to the Bayesian posterior of the hidden variable θ. After presenting a batch of 50 periods, the error values were calculated, and their means were used to update the weights and biases.

The initial values of internal connectivity J_{ij} were drawn from a Gaussian distribution with SD $g/\sqrt{N}$, where we set $g = 8.0$ in Figs. 2, 3*A*, and 4 and *SI Appendix*, Figs. S2 *D–F* and S3 and $g = 1.05$ in the other cases unless it is stated otherwise. The readout weights W_{ij} were initially drawn from a Gaussian distribution with SD $1/\sqrt{N}$. We assumed the zero readout biases $b_i = 0$ for all network neurons through this task. The input connections K_{ij} from sensory neurons to the recurrent networks were drawn from a standard normal distribution (10 times larger deviations are used in *SI Appendix*, Fig. S3), and their values were fixed in simulations.

Temporal Statistical Task. To study the capacity of the chaotic networks to capture temporal relations by their dynamics, we considered a temporal task in which the recurrent neural networks receive inputs from sensory neurons and generate their outputs so that their sequence follows target transient probabilities.

We considered two groups of sensory neurons A and B. The recurrent neural networks have $N = 100$ neurons as seen in Fig. 6*A*. Both sensory neurons take binary states and we trained network dynamics for three of their firing patterns: $\left(x^{\mathrm{A}}, x^{\mathrm{B}}\right) = (0, 0), (1, 0), (0, 1)$, as spontaneous, A-evoked, B-evoked states, respectively. For each pattern, we set the target transition probabilities that the network should learn, as represented with the red blank boxes in Fig. 6*C*; the target transition probability matrices are denoted concretely as $(1/3, 1/3, 1/3; 1/3, 1/3, 1/3; 1/3, 1/3, 1/3)$ for $(x^{\mathrm{A}}, x^{\mathrm{B}}) = (0, 0)$, $(0.25, 0.5, 0.25; 0.25, 0.25, 0.5; 0.5, 0.25, 0.25)$ for $(x^{\mathrm{A}}, x^{\mathrm{B}}) = (1, 0)$, and $(0.3, 0.1, 0.6; 0.6, 0.3, 0.1; 0.1, 0.6, 0.3)$ for $(x^{\mathrm{A}}, x^{\mathrm{B}}) = (0, 1)$, where the entry of ith row and jth column denotes the conditional probability $p(\theta(t+1) = i|\theta(t) = j)$. We trained the network with a time window of 500 time steps for each of the three input patterns.

The gain parameter of synaptic connectivity was set as $g = 12.5$, the initial K_{ij} was drawn from a standard Gaussian distribution, and initial W_{ij} was a Gaussian with a SD $1/\sqrt{N}$. We initially set biases of neurons and readout weights to zero and assumed a nonzero plastic baseline parameter $\mathbf{c}$ for neural activity.

Data, Materials, and Software Availability. Codes reproducing the main figures can be found at GitHub (https://github.com/yu-terada/chaos-sampling) (103). All other data are included in the manuscript and/or *SI Appendix*.

ACKNOWLEDGMENTS. We thank Yonatan Aljadeff for providing critical comments on the manuscript, which improved the paper significantly. This work was supported by RIKEN Center for Brain Science, the Special Postdoctoral Research Program at RIKEN (Y.T.), Brain/MINDS from AMED under Grant No. JP15dm0207001 (T.T.), MEXT KAKENHI Grant Nos. JP18H05432 (T.T.) and JP19K20365 (Y.T.), the CRCNS award by the US Department of Energy: DE-SC0022042 (Y.T.), and the JST CREST program JPMJCR23N2 (T.T.).

Author affiliations: [a]Laboratory for Neural Computation and Adaptation, RIKEN Center for Brain Science, Saitama 351-0198, Japan; [b]Department of Neurobiology, University of California, San Diego, La Jolla, CA 92093; [c]The Institute for Physics of Intelligence, The University of Tokyo, Tokyo 113-0033, Japan; and [d]Department of Mathematical Informatics, Graduate School of Information Science and Technology, The University of Tokyo, Tokyo 113-8656, Japan

1. M. O. Ernst, M. S. Banks, Humans integrate visual and haptic information in a statistically optimal fashion. *Nature* **415**, 429–433 (2002).
2. D. C. Knill, A. Pouget, The Bayesian brain: The role of uncertainty in neural coding and computation. *Trends Neurosci.* **27**, 712–719 (2004).
3. K. P. Körding, D. M. Wolpert, Bayesian integration in sensorimotor learning. *Nature* **427**, 244–247 (2004).
4. T. Yang, M. N. Shadlen, Probabilistic reasoning by neurons. *Nature* **447**, 1075–1080 (2007).
5. P. Berkes, G. Orbán, M. Lengyel, J. Fiser, Spontaneous cortical activity reveals hallmarks of an optimal internal model of the environment. *Science* **331**, 83–87 (2011).
6. H. Sohn, D. Narain, N. Meirhaeghe, M. Jazayeri, Bayesian computation through cortical latent dynamics. *Neuron* **103**, 934–947 (2019).
7. W. J. Ma, J. M. Beck, P. E. Latham, A. Pouget, Bayesian inference with probabilistic population codes. *Nat. Neurosci.* **9**, 1432–1438 (2006).
8. P. Hoyer, A. Hyvärinen, Interpreting neural response variability as Monte Carlo sampling of the posterior. *Adv. Neural Inf. Process. Syst.* **15**, 293–300 (2002).
9. G. Orbán, P. Berkes, J. Fiser, M. Lengyel, Neural variability and sampling-based probabilistic representations in the visual cortex. *Neuron* **92**, 530–543 (2016).
10. L. Buesing, J. Bill, B. Nessler, W. Maass, Neural dynamics as sampling: A model for stochastic computation in recurrent networks of spiking neurons. *PLoS Comput. Biol.* **7**, e1002211 (2011).
11. D. Pecevski, L. Buesing, W. Maass, Probabilistic inference in general graphical models through sampling in stochastic networks of spiking neurons. *PLoS Comput. Biol.* **7**, e1002294 (2011).

12. L. Aitchison, M. Lengyel, The Hamiltonian brain: Efficient probabilistic inference with excitatory-inhibitory neural circuit dynamics. *PLoS Comput. Biol.* **12**, e1005186 (2016).
13. O. J. Hénaff, Z. M. Boundy-Singer, K. Meding, C. M. Ziemba, R. L. Goris, Representation of visual uncertainty through neural gain variability. *Nat. Commun.* **11**, 1–12 (2020).
14. R. Echeveste, L. Aitchison, G. Hennequin, M. Lengyel, Cortical-like dynamics in recurrent circuits optimized for sampling-based probabilistic inference. *Nat. Neurosci.* **23**, 1138–1149 (2020).
15. D. Festa, A. Aschner, A. Davila, A. Kohn, R. Coen-Cagli, Neuronal variability reflects probabilistic inference tuned to natural image statistics. *Nat. Commun.* **12**, 1–11 (2021).
16. L. Aitchison, G. Hennequin, M. Lengyel, Sampling-based probabilistic inference emerges from learning in neural circuits with a cost on reliability. arXiv [Preprint] (2018). https://doi.org/10.48550/arXiv.1807.08952 (Accessed 28 July 2023).
17. J. Malkin, C. O'Donnell, C. Houghton, L. Aitchison, Signatures of Bayesian inference emerge from energy efficient synapses. arXiv [Preprint] (2023). https://doi.org/10.48550/arXiv.2309.03194 (Accessed 28 July 2023).
18. C. Savin, S. Denève, Spatio-temporal representations of uncertainty in spiking neural networks. *Adv. Neural Inf. Proc. Sys.* **27**, 2024–2032 (2014).
19. Y. Qi, P. Gong, Fractional neural sampling as a theory of spatiotemporal probabilistic computations in neural circuits. *Nat. Commun.* **13**, 4572 (2022).
20. W. J. Ma, K. P. Kording, D. Goldreich, *Bayesian Models of Perception and Action: An Introduction* (MIT Press, 2023).
21. W. H. Zhang, S. Wu, K. Josić, B. Doiron, Sampling-based Bayesian inference in recurrent circuits of stochastic spiking neurons. *Nat. Commun.* **14**, 7074 (2023).
22. A. Luczak, P. Barthó, S. L. Marguet, G. Buzsáki, K. D. Harris, Sequential structure of neocortical spontaneous activity in vivo. *Proc. Natl Acad. Sci. U.S.A.* **104**, 347–352 (2007).
23. L. Avitan, C. Stringer, Not so spontaneous: Multi-dimensional representations of behaviors and context in sensory areas. *Neuron* **110**, 3064–3075 (2022).
24. G. J. Tomko, D. R. Crapper, Neuronal variability: Non-stationary responses to identical visual stimuli. *Brain Res.* **79**, 405–418 (1974).
25. M. N. Shadlen, W. T. Newsome, The variable discharge of cortical neurons: Implications for connectivity, computation, and information coding. *J. Neurosci.* **18**, 3870–3896 (1998).
26. M. M. Churchland *et al*., Stimulus onset quenches neural variability: A widespread cortical phenomenon. *Nat. Neurosci.* **13**, 369–378 (2010).
27. M. E. Raichle *et al*., A default mode of brain function. *Proc. Natl Acad. Sci. U.S.A.* **98**, 676–682 (2001).
28. Y. Yeshurun, M. Nguyen, U. Hasson, The default mode network: Where the idiosyncratic self meets the shared social world. *Nat. Rev. Neurosci.* **22**, 181–192 (2021).
29. D. Plenz, T. C. Thiagarajan, The organizing principles of neuronal avalanches: Cell assemblies in the cortex? *Trends Neurosci.* **30**, 101–110 (2007).
30. J. M. Beck, W. J. Ma, X. Pitkow, P. E. Latham, A. Pouget, Not noisy, just wrong: The role of suboptimal inference in behavioral variability. *Neuron* **74**, 30–39 (2012).
31. M. London, A. Roth, L. Beeren, M. Häusser, P. E. Latham, Sensitivity to perturbations in vivo implies high noise and suggests rate coding in cortex. *Nature* **466**, 123–127 (2010).
32. M. M. Churchland, L. Abbott, Two layers of neural variability. *Nat. Neurosci.* **15**, 1472–1474 (2012).
33. R. B. Stein, E. R. Gossen, K. E. Jones, Neuronal variability: Noise or part of the signal? *Nat. Rev. Neurosci.* **6**, 389–397 (2005).
34. A. A. Faisal, L. P. Selen, D. M. Wolpert, Noise in the nervous system. *Nat. Rev. Neurosci.* **9**, 292–303 (2008).
35. E. T. Rolls, G. Deco, *The Noisy Brain: Stochastic Dynamics as a Principle of Brain Function* (Oxford University Press, 2010), vol. 34.
36. W. Gerstner, W. M. Kistler, R. Naud, L. Paninski, *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition* (Cambridge University Press, 2014).
37. J. G. G. Borst, The low synaptic release probability in vivo. *Trends Neurosci.* **33**, 259–266 (2010).
38. C. Van Vreeswijk, H. Sompolinsky, Chaos in neuronal networks with balanced excitatory and inhibitory activity. *Science* **274**, 1724–1726 (1996).
39. H. Sompolinsky, A. Crisanti, H. J. Sommers, Chaos in random neural networks. *Phys. Rev. Lett.* **61**, 259 (1988).
40. A. Renart *et al*., The asynchronous state in cortical circuits. *Science* **327**, 587–590 (2010).
41. J. Kadmon, H. Sompolinsky, Transition to chaos in random neuronal networks. *Phys. Rev. X* **5**, 041030 (2015).
42. J. Aljadeff, M. Stern, T. Sharpee, Transition to chaos in random networks with cell-type-specific connectivity. *Phys. Rev. Lett.* **114**, 088101 (2015).
43. Ł. Kuśmierz, S. Ogawa, T. Toyoizumi, Edge of chaos and avalanches in neural networks with heavy-tailed synaptic weight distribution. *Phys. Rev. Lett.* **125**, 028101 (2020).
44. K. Krishnamurthy, T. Can, D. J. Schwab, Theory of gating in recurrent neural networks. *Phys. Rev. X* **12**, 011011 (2022).
45. R. Engelken, F. Wolf, L. F. Abbott, Lyapunov spectra of chaotic recurrent neural networks. *Phys. Rev. Res.* **5**, 043044 (2023).
46. N. Bertschinger, T. Natschläger, Real-time computation at the edge of chaos in recurrent neural networks. *Neural Comput.* **16**, 1413–1436 (2004).
47. R. Legenstein, W. Maass, Edge of chaos and prediction of computational performance for neural circuit models. *Neural Netw.* **20**, 323–334 (2007).
48. T. Toyoizumi, L. Abbott, Beyond the edge of chaos: Amplification and temporal integration by recurrent networks in the chaotic regime. *Phys. Rev. E* **84**, 051908 (2011).
49. G. Yang, S. Schoenholz, Mean field residual networks: On the edge of chaos. *Adv. Neural Inf. Process. Syst.* **30**, 7103–7114 (2017).
50. D. Sussillo, L. F. Abbott, Generating coherent patterns of activity from chaotic neural networks. *Neuron* **63**, 544–557 (2009).
51. R. Laje, D. V. Buonomano, Robust timing and motor patterns by taming chaos in recurrent neural networks. *Nat. Neurosci.* **16**, 925–933 (2013).
52. G. M. Hoerzer, R. Legenstein, W. Maass, Emergence of complex computational structures from chaotic neural networks through reward-modulated Hebbian learning. *Cereb. Cortex* **24**, 677–690 (2014).
53. O. Barak, D. Sussillo, R. Romo, M. Tsodyks, L. Abbott, From fixed points to chaos: Three models of delayed discrimination. *Prog. Neurobiol.* **103**, 214–222 (2013).
54. M. Farrell, S. Recanatesi, T. Moore, G. Lajoie, E. Shea-Brown, Gradient-based learning drives robust representations in recurrent neural networks by balancing compression and expansion. *Nat. Mach. Intell.* **4**, 564–573 (2022).
55. T. Can, K. Krishnamurthy, Emergence of memory manifolds. arXiv [Preprint] (2021). https://doi.org/10.48550/arXiv.2109.03879 (Accessed 28 July 2023).
56. R. Salakhutdinov, G. Hinton, "Deep Boltzmann machines" in *AISTATS* (Proceedings of Machine Learning Research, 2009), pp. 448–455.
57. D. P. Kingma, M. Welling, Auto-encoding variational Bayes. arXiv [Preprint] (2013). https://doi.org/10.48550/arXiv.1312.6114 (Accessed 28 July 2023).
58. I. Goodfellow *et al*., Generative adversarial nets. *Adv. Neural Inf. Process. Syst.* **27**, 2672–2680 (2014).
59. J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models. *Adv. Neural Inf. Process. Syst.* **33**, 6840–6851 (2020).
60. A. G. Barto, P. Anandan, Pattern-recognizing stochastic learning automata. *IEEE Trans. Syst. Man Cybern.*, 360–375 (1985).
61. Y. Le Cun, C. Galland, G. E. Hinton, Gemini: Gradient estimation through matrix inversion after noise injection. *Adv. Neural Inf. Process. Syst.* **1**, 141–148 (1988).
62. I. R. Fiete, H. S. Seung, Gradient learning in spiking neural networks by dynamic perturbation of conductances. *Phys. Rev. Lett.* **97**, 048104 (2006).
63. N. Hiratani, Y. Mehta, T. Lillicrap, P. E. Latham, On the stability and scalability of node perturbation learning. *Adv. Neural Information. Proc. Syst.* **35**, 31929–31941 (2022).
64. H. F. Song, G. R. Yang, X. J. Wang, Reward-based training of recurrent neural networks for cognitive and value-based tasks. *eLife* **6**, e21492 (2017).
65. R. Legenstein, S. M. Chase, A. B. Schwartz, W. Maass, A reward-modulated Hebbian learning rule can explain experimentally observed network reorganization in a brain control task. *J. Neurosci.* **30**, 8400–8410 (2010).
66. T. Miconi, Biologically plausible learning in recurrent neural networks reproduces neural dynamics observed during cognitive tasks. *eLife* **6**, e20899 (2017).
67. J. M. Murray, Local online learning in recurrent networks with random feedback. *eLife* **8**, e43299 (2019).
68. Ł Kuśmierz, T. Isomura, T. Toyoizumi, Learning with three factors: Modulating Hebbian plasticity with errors. *Curr. Opin. Neurobiol.* **46**, 170–177 (2017).
69. W. Gerstner, M. Lehmann, V. Liakoni, D. Corneil, J. Brea, Eligibility traces and plasticity on behavioral time scales: Experimental support of neoHebbian three-factor learning rules. *Front. Neural Circuits* **12**, 53 (2018).
70. T. P. Lillicrap, A. Santoro, L. Marris, C. J. Akerman, G. Hinton, Backpropagation and the brain. *Nat. Rev. Neurosci.* **21**, 335–346 (2020).
71. K. Doya, S. Ishii, A. Pouget, R. P. Rao, *Bayesian Brain: Probabilistic Approaches to Neural Coding* (MIT Press, 2007).
72. J. Trommershauser, K. Kording, M. S. Landy, *Sensory Cue Integration* (Oxford University Press, 2011).
73. W. Maass, On the computational power of winner-take-all. *Neural Comput.* **12**, 2519–2535 (2000).
74. E. Hellinger, Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen. *J. Reine Angew. Math.* **1909**, 210–271 (1909).
75. L. Molgedey, J. Schuchhardt, H. G. Schuster, Suppressing chaos in neural networks by noise. *Phys. Rev. Lett.* **69**, 3717 (1992).
76. T. Kurikawa, O. Barak, K. Kaneko, Repeated sequential learning increases memory capacity via effective decorrelation in a recurrent neural network. *Phys. Rev. Res.* **2**, 023307 (2020).
77. M. N. Shadlen, D. Shohamy, Decision making and sequential sampling from memory. *Neuron* **90**, 927–939 (2016).
78. B. B. Scott *et al*., Imaging cortical dynamics in GCaMP transgenic rats with a head-mounted widefield macroscope. *Neuron* **100**, 1045–1058 (2018).
79. K. Ota *et al*., Fast, cell-resolution, contiguous-wide two-photon imaging to reveal functional network architectures across multi-modal cortical areas. *Neuron* **109**, 1810–1824 (2021).
80. N. A. Steinmetz *et al*., Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings. *Science* **372**, eabf4588 (2021).
81. A. M. Packer, B. Roska, M. Häusser, Targeting neurons and photons for optogenetics. *Nat. Neurosci.* **16**, 805–815 (2013).
82. B. B. Ujfalussy, G. Orbán, Sampling motion trajectories during hippocampal theta sequences. *eLife* **11**, e74058 (2022).
83. S. Dasgupta, I. Nishikawa, A. Kazuyuki, T. Toyoizumi, "Efficient signal processing in random networks that generate variability: A comparison of internally generated and externally induced variability" in *NIPS Workshop on Modelling Inference for Dynamics on Complex Interact Networks* (2015).
84. C. Chen, A. Abbott, D. Stilwell, "Multi-level generative chaotic recurrent network for image inpainting" in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (2021), pp. 3626–3635.
85. J. Jordan *et al*., Deterministic networks for probabilistic computing. *Sci. Rep.* **9**, 1–17 (2019).
86. T. Matsuki, K. Shibata, Adaptive balancing of exploration and exploitation around the edge of chaos in internal-chaos-based learning. *Neural Netw.* **132**, 19–29 (2020).
87. J. Schuecker, S. Goedeke, M. Helias, Optimal sequence memory in driven random networks. *Phys. Rev. X* **8**, 041029 (2018).
88. S. Ogawa, F. Fumarola, L. Mazzucato, Baseline control of optimal performance in recurrent neural networks. bioRxiv [Preprint] (2022). https://doi.org/10.1101/2022.05.11.491436 (Accessed 28 July 2023).
89. K. Rajan, C. D. Harvey, D. W. Tank, Recurrent network models of sequence generation and memory. *Neuron* **90**, 128–142 (2016).
90. G. R. Yang, M. R. Joglekar, H. F. Song, W. T. Newsome, X. J. Wang, Task representations in neural networks trained to perform many cognitive tasks. *Nat. Neurosci.* **22**, 297–306 (2019).
91. S. Saxena, A. A. Russo, J. Cunningham, M. M. Churchland, Motor cortex activity across movement speeds is predicted by network-level strategies for generating muscle activity. *eLife* **11**, e67620 (2022).

92. A. Dubreuil, A. Valente, M. Beiran, F. Mastrogiuseppe, S. Ostojic, The role of population structure in computations through neural dynamics. *Nat. Neurosci.* **25**, 783–794 (2022).
93. T. Kurikawa, K. Kaneko, Multiple-timescale neural networks: generation of history-dependent sequences and inference through autonomous bifurcations. *Front. Comput. Neurosci.* **15**, 743537 (2021).
94. M. Stern, N. Istrate, L. Mazzucato, A reservoir of timescales in random neural networks. bioRxiv [Preprint] (2021). https://doi.org/10.1101/2021.10.11.463861.
95. A. Rubin, M. M. Yartsev, N. Ulanovsky, Encoding of head direction by hippocampal place cells in bats. *J. Neurosci.* **34**, 1067–1080 (2014).
96. Si. Amari, Dynamics of pattern formation in lateral-inhibition type neural fields. *Biol. Cybern.* **27**, 77–87 (1977).
97. R. Ben-Yishai, R. L. Bar-Or, H. Sompolinsky, Theory of orientation tuning in visual cortex. *Proc. Natl Acad. Sci. U.S.A.* **92**, 3844–3848 (1995).
98. G. Lajoie, K. K. Lin, J. P. Thivierge, E. Shea-Brown, Encoding in balanced networks: Revisiting spike patterns and chaos in stimulus-driven systems. *PLoS Comput. Biol.* **12**, e1005258 (2016).
99. T. Haruna, K. Nakajima, Optimal short-term memory before the edge of chaos in driven random recurrent networks. *Phys. Rev. E* **100**, 062312 (2019).
100. R. J. Williams, Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.* **8**, 229–256 (1992).
101. J. Werfel, X. Xie, H. S. Seung, Learning curves for stochastic gradient descent in linear feedforward networks. *Neural Comput.* **17**, 2699–2718 (2005).
102. D. P. Kingma, J. Ba, Adam: A method for stochastic optimization. arXiv [Preprint] (2014). https://doi.org/10.48550/arXiv.1412.6980 (Accessed 28 July 2023).
103. Y. Terada, T. Toyoizumi, Chaos sampling by recurrent neural networks. Github. https://github.com/yu-terada/chaos-sampling. Deposited 6 April 2024.