Reward-based training of recurrent neural networks for cognitive and value-based tasks

H. Francis Song[1], Guangyu R. Yang[1], & Xiao-Jing Wang[1,2,*]

[1] Center for Neural Science, New York University, New York, NY 10003, USA

[2] NYU-ECNU Institute of Brain and Cognitive Science, NYU Shanghai, Shanghai 200122, China

[*] For correspondence: xjwang@nyu.edu

Abstract

Trained neural network models, which exhibit many features observed in neural recordings from behaving animals and whose activity and connectivity can be fully analyzed, may provide insights into neural mechanisms. In contrast to commonly used methods for supervised learning from graded error signals, however, animals learn from reward feedback on definite actions through reinforcement learning. Reward maximization is particularly relevant when the optimal behavior depends on an animal's internal judgment of confidence or subjective preferences. Here, we describe reward-based training of recurrent neural networks in which a value network guides learning by using the selected actions and activity of the policy network to predict future reward. We show that such models capture both behavioral and electrophysiological findings from well-known experimental paradigms. Our results provide a unified framework for investigating diverse cognitive and value-based computations, including a role for value representation that is essential for learning, but not executing, a task.

Introduction

A major challenge in uncovering the neural mechanisms underlying complex behavior is our incomplete access to relevant circuits in the brain. Recent work has shown that model neural networks optimized for a wide range of tasks, including visual object recognition (Cadieu et al., 2014; Yamins et al., 2014; Hong et al., 2016), perceptual decision-making and working memory (Mante et al., 2013; Barak et al., 2013; Carnevale et al., 2015; Song et al., 2016; Miconi, 2016), timing and sequence generation (Laje & Buonomano, 2013; Rajan et al., 2015), and motor reach (Hennequin et al., 2014; Sussillo et al., 2015), can reproduce important features of neural activity recorded in numerous cortical areas of behaving animals. The analysis of such circuits, whose activity and connectivity are fully known, has therefore re-emerged as a promising tool for understanding neural computation (Zipser & Andersen, 1988; Sussillo, 2014; Gao & Ganguli, 2015). Constraining network training with tasks for which detailed neural recordings are available may also provide insights into the principles that govern learning in biological circuits (Sussillo et al., 2015; Song et al., 2016).

Previous applications of this approach to “cognitive-type” behavior such as perceptual decision-making and working memory have focused on supervised learning from graded error signals. Animals, however, learn to perform specific tasks from reward feedback provided by the experimentalist in response to definite actions, i.e., through reinforcement learning (Sutton & Barto, 1998). Unlike in supervised learning where the network must be given the correct response on each trial, reinforcement learning merely provides evaluative feedback to the network, i.e., whether the selected action was good or bad. For the purposes of using model neural networks to generate hypotheses about neural mechanisms, this is particularly relevant in tasks where the optimal behavior depends on an animal’s internal state or subjective preferences. In a perceptual decision-making task with postdecision wagering, for example, on some trials the animal can opt for a “sure” choice that results in a small (compared to the correct choice) but certain reward (Kiani & Shadlen, 2009). The optimal decision regarding whether or not to select the sure choice depends not only on the task condition, such as the proportion of coherently moving dots, but also on the animal’s own confidence in its decision *during the trial*. Learning to make this judgment cannot be reduced to minimizing a trial-by-trial error signal; it can be learned, however, by choosing the actions that result in greatest reward. Meanwhile, supervised learning is inherently unsuitable for value-based, or economic, decision-making where the "correct" judgment depends explicitly on rewards associated with different actions, even for identical sensory inputs (Padoa-Schioppa & Assad, 2006). More fundamentally, reward plays a central role in all types of animal learning (Sugrue et al., 2005). Explicitly incorporating reward into network training is therefore a necessary step toward elucidating the biological substrates of learning, in particular reward-dependent synaptic plasticity (Seung, 2003; Soltani et al., 2006; Izhikevich, 2007; Urbanczik & Senn, 2009; Frémaux et al., 2010; Soltani & Wang, 2010; Hoerzer et al., 2014; Brosch et al., 2015; Friedrich & Lengyel, 2016) and the role of different brain structures in learning (Frank & Claus, 2006).

In this work, we build on advances in policy gradient reinforcement learning, specifically the REINFORCE algorithm (Williams, 1992; Sutton et al., 2000; Peters & Schaal, 2008; Wierstra et al., 2009), to demonstrate reward-based training of recurrent neural networks (RNNs) for several well-known experimental paradigms in systems neuroscience. The networks consist of two modules in an "actor-critic" architecture (Barto et al., 1983; Grondman et al., 2012), in which a policy network uses inputs provided by the environment to select actions that maximize reward, while a value network uses the selected actions and activity of the policy network to predict future reward and guide learning. We first present networks trained for tasks that have been studied previously using various forms of supervised learning (Mante et al., 2013; Barak et al., 2013; Song et al., 2016); they are characterized by "simple" input-output mappings in which the correct response for each trial depends only on the task condition, and include perceptual decision-making, context-dependent integration, multisensory integration, and parametric working memory tasks. We then show results for tasks in which the optimal behavior depends on the animal's internal judgment of confidence or subjective preferences, specifically a perceptual decision-making task with postdecision wagering (Kiani & Shadlen, 2009) and a value-based economic choice task

(Padoa-Schioppa & Assad, 2006). Interestingly, unlike for the other tasks where we focus on comparing the activity of units in the policy network to neural recordings in the prefrontal and posterior parietal cortex of animals performing the same tasks, for the economic choice task the activity of the value network exhibits a striking resemblance to neural recordings from the orbitofrontal cortex (OFC), which has long been implicated in the representation of reward-related signals (Wallis, 2007).

Indeed, a crucial feature of our REINFORCE-based model is that a reward baseline—in this case a recurrently connected value network—is essential for learning, but not for executing the task, because the latter depends only on the policy network. Importantly, learning can still occur without the value network but is much more difficult. It is sometimes observed in experiments that reward-modulated structures in the brain such as the basal ganglia or orbitofrontal cortex are necessary for learning or adapting to a changing environment, but not for executing a previously learned skill (Turner & Desmurget, 2010; Schoenbaum et al., 2011; Stalnaker et al., 2015). This suggests that one possible role for such circuits may be representing an accurate baseline to guide learning. Moreover, since confidence is closely related to expected reward in many cognitive tasks, the explicit computation of expected reward by the value network provides a concrete, learning-based rationale for confidence estimation as a fundamental and ubiquitous component of decision-making (Kepecs et al., 2008; Wei & Wang, 2015), even when it is not strictly required for performing the task.

Conceptually, the formulation of behavioral tasks in the language of reinforcement learning presented here is closely related to the solution of partially observable Markov decision processes (POMDPs) (Kaelbling et al., 1998) using either model-based belief states (Rao, 2010) or model-free working memory (Todd et al., 2008). Indeed, as in Dayan & Daw (2008) one of the goals of this work is to unify related computations into a common language that is applicable to a wide range of tasks in systems neuroscience. However, the formulation using policies represented by RNNs allows for a far more general description, and, in particular, makes the assumption of a Markovian environment unnecessary (Wierstra et al., 2009). Such policies can also be compared more directly to "optimal" solutions when they are known, for instance to the signal detection theory account of perceptual decision-making (Gold & Shadlen, 2007). Thus, in addition to expanding the range of tasks and neural mechanisms that can be studied with trained RNNs, our work provides a convenient framework for the study of cognitive and value-based computations in the brain, which have often been viewed from distinct perspectives but in fact arise from the same reinforcement learning paradigm.

Results

Policy gradient reinforcement learning for behavioral tasks

For concreteness, we illustrate the following in the context of a simplified perceptual decision-making task based on the random dots motion (RDM) discrimination task as described in Kiani et al. (2008) (Figure 1A). In its simplest form, in an RDM task the monkey must maintain fixation until a "go" cue indicates that the monkey should make a decision regarding the direction of coherently moving dots on the screen. Thus the three possible actions available to the monkey at any given time are fixate, choose left, or choose right. The true direction of motion, which can be considered a *state* of the environment, is not known to the monkey with certainty, i.e., is *partially observable*. The monkey must therefore use the noisy sensory evidence to infer the direction in order to select the correct response at the end of the trial. Breaking fixation early results in a negative reward in the form of a timeout, while giving the correct response after the fixation cue is extinguished results in a positive reward in the form of juice. The goal of this section is to give a general description of such tasks and how an animal or RNN can learn a behavioral *policy* for choosing actions at each time to maximize the reward.

Consider a typical interaction between an experimentalist and animal, which we more generally call the environment $\mathscr{E}$ and agent $\mathscr{A}$, respectively (Figure 1B). At each time t the agent chooses to perform actions $\mathbf{a}_t$ after observing inputs $\mathbf{u}_t$ provided by the environment, and the probability of choosing actions $\mathbf{a}_t$ is given by the agent's policy $\pi_\theta(\mathbf{a}_t|\mathbf{u}_{1:t})$ with parameters θ. Here the policy is implemented as an RNN, so that θ comprises the connection weights, biases, and initial state of the network. The policy at time t can depend on all past and current inputs $\mathbf{u}_{1:t} = (\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_t)$, allowing the agent to integrate sensory evidence or use working memory to perform the task. The exception is at $t = 0$, when the agent has yet to interact with the environment and selects its actions "spontaneously" according to $\pi_\theta(\mathbf{a}_0)$. We note that, if the inputs give exact information about the environmental state, i.e., if $\mathbf{u}_t = \mathbf{s}_t$, then the task becomes a Markov decision process. In general, however, the inputs only provide partial information about the environmental states, requiring the network to accumulate evidence over time to determine the state of the world. In this work we only consider cases where the agent chooses one out of N_a possible actions at each time, so that $\pi_\theta(\mathbf{a}_t|\mathbf{u}_{1:t})$ for each t is a discrete, normalized probability distribution over the possible actions $a_1, \dots, a_{N_a}$. More generally, $\mathbf{a}_t$ can implement several distinct actions or even continuous actions by representing, for example, the means of Gaussian distributions (Peters & Schaal, 2008; Wierstra et al., 2009). After each set of actions by the agent at time t the environment provides a reward (or special observable) ρ_{t+1} at time $t+1$, which the agent attempts to maximize in the sense described below.

In the case of the example RDM task above (Figure 1A), the environment provides (and the agent receives) as inputs a fixation cue and noisy evidence for two choices L and R during a variable-length stimulus presentation period. The strength of evidence, or the difference between the evidence for L and R, is called the coherence, and in the actual RDM experiment corresponds to the percentage of dots moving coherently in one direction on the screen. The agent chooses to perform one of $N_a = 3$ actions at each time: fixate ($a_t = \mathrm{F}$), choose L ($a_t = \mathrm{L}$), or choose R ($a_t = \mathrm{R}$). In this task, the agent must choose F as long as the fixation cue is on, and then, when the fixation cue is turned off to indicate that the agent should make a decision, correctly choose L or R depending on the sensory evidence. Indeed, for all tasks in this work we required that the network "make a decision" (i.e., break fixation to indicate a choice at the appropriate time) on at least 99% of the trials, whether the response was correct or not. A trial ends when the agent chooses L or R regardless of the task epoch: breaking fixation early before the go cue results in an aborted trial and a negative reward $\rho_t = -1$, while a correct decision is rewarded with $\rho_t = +1$. Making the wrong decision results in no reward, $\rho_t = 0$. For the zero-coherence condition the agent is rewarded randomly on half the trials regardless of its choice. Otherwise the reward is always $\rho_t = 0$.

Formally, a trial proceeds as follows. At time $t = 0$, the environment is in state $\mathbf{s}_0$ with probability $\mathscr{E}(\mathbf{s}_0)$. The state $\mathbf{s}_0$ can be considered the starting time (i.e., $t = 0$) and "task condition," which in the RDM example consists of the direction of motion of the dots (i.e., whether the correct response is L or R) and the coherence of the dots (the difference between evidence for L and R). The time component of the state, which is updated at each step, allows the environment to present different inputs to the agent depending on the task epoch. The true state $\mathbf{s}_0$ (such as the direction of the dots) is only partially observable to the agent, so that the agent must instead infer the state through inputs $\mathbf{u}_t$ provided by the environment during the course of the trial. As noted previously, the agent initially chooses actions $\mathbf{a}_0$ with probability $\pi_\theta(\mathbf{a}_0)$. The networks in this work almost always begin by choosing F, or fixation.

At time $t = 1$, the environment, depending on its previous state $\mathbf{s}_0$ and the agent's action $\mathbf{a}_0$, transitions to state $\mathbf{s}_1$ with probability $\mathscr{E}(\mathbf{s}_1|\mathbf{s}_0, \mathbf{a}_0)$ and generates reward ρ_1. In the perceptual decision-making example, only the time advances since the trial condition remains constant throughout. From this state the environment generates observable $\mathbf{u}_1$ with a distribution given by $\mathscr{E}(\mathbf{u}_1|\mathbf{s}_1)$. If $t = 1$ were in the stimulus presentation period, for example, $\mathbf{u}_1$ would provide noisy evidence for L or R, as well as the fixation cue. In response, the agent, depending on the inputs $\mathbf{u}_1$ it receives from the environment, chooses actions $\mathbf{a}_1$ with probability $\pi_\theta(\mathbf{a}_1|\mathbf{u}_{1:1}) = \pi_\theta(\mathbf{a}_1|\mathbf{u}_1)$. The environment, depending on its previous states $\mathbf{s}_{0:1} = (\mathbf{s}_0, \mathbf{s}_1)$ and the agent's previous actions $a_{0:1} = (\mathbf{a}_0, \mathbf{a}_1)$, then transitions to state $\mathbf{s}_2$ with probability $\mathscr{E}(\mathbf{s}_2|\mathbf{s}_{0:1}, \mathbf{a}_{0:1})$ and generates reward ρ_2. These steps are repeated until the end of the trial at time T. Trials can terminate before T (e.g., for breaking fixation early), so that T represents the maximum length of a trial. In order to emphasize that rewards follow actions, we adopt the convention in which the agent performs actions at $t = 0, \ldots, T$ and receives rewards at $t = 1, \ldots, T + 1$.

The goal of the agent is to maximize the sum of expected future rewards at time $t = 0$, or expected return

$$J(\theta) = \mathbb{E}_H \left[\sum_{t=0}^{T} \rho_{t+1} \right], \tag{1}$$

where the expectation $\mathbb{E}_H$ is taken over all possible trial histories $H = (\mathbf{s}_{0:T+1}, \mathbf{u}_{1:T}, \mathbf{a}_{0:T})$ consisting of the states of the environment, the inputs given to the agent, and the actions of the agent. In practice, the expectation value in Equation 1 is estimated by performing N_{trials} trials for each policy update, i.e., with a Monte Carlo approximation. Here we only consider the episodic, "finite-horizon" case where T is finite. The expected return depends on the policy and hence parameters θ, and we use Adam stochastic gradient descent (SGD) (Kingma & Ba, 2015) with gradient clipping (Graves, 2013; Pascanu et al., 2013b) to find the parameters that maximize this reward (Methods).

More specifically, for the policy network we use gradient descent to minimize an objective function $\mathscr{L}^{\pi}$ of the form

$$\mathscr{L}^{\pi}(\theta) = \frac{1}{N_{\text{trials}}} \sum_{n=1}^{N_{\text{trials}}} \left[-J_n(\theta) + \Omega_n^{\pi}(\theta) \right] \tag{2}$$

with respect to the connection weights, biases, and initial state of the policy network, which we collectively denote as θ. Here $\Omega_n^{\pi}(\theta)$ contains any regularization terms for the policy network, for instance an entropy term to control the degree of exploration (Xu et al., 2015). The key gradient $\nabla_\theta J_n(\theta)$ is given for each trial n by the REINFORCE algorithm (Williams, 1992; Sutton et al., 2000; Peters & Schaal, 2008; Wierstra et al., 2009) as

$$\nabla_\theta J_n(\theta) = \sum_{t=0}^{T} \left[\nabla_\theta \log \pi_\theta(\mathbf{a}_t | \mathbf{u}_{1:t}) \right] \left[\sum_{\tau=t}^{T} \rho_{\tau+1} - v_\phi(\mathbf{a}_{1:t}, \mathbf{r}_{1:t}^{\pi}) \right], \tag{3}$$

where $\mathbf{r}_{1:t}^{\pi}$ are the firing rates of the policy network units up to time t, v_ϕ denotes the value function as described below, and the gradient $\nabla_\theta \log \pi_\theta(\mathbf{a}_t | \mathbf{u}_{1:t})$, known as the *eligibility*, [and likewise $\nabla_\theta \Omega_n^{\pi}(\theta)$] is computed by backpropagation through time (BPTT) (Rumelhart et al.,

1986) for the selected actions $\mathbf{a}_t$. The sum over rewards in large brackets only runs over $\tau = t, \ldots, T$, which reflects the fact that actions do not affect past rewards. In this form the terms in the gradient have the intuitive property that they are nonzero only if the actual return deviates from what was predicted by the baseline. This reward baseline is an important feature in the success of almost all REINFORCE-based algorithms, and is here represented by a second RNN v_ϕ with parameters ϕ in addition to the policy network π_θ. This baseline network, which we call the *value network*, uses the selected actions $\mathbf{a}_{1:t}$ and activity of the policy network $\mathbf{r}^\pi_{1:t}$ to predict the expected return at each time $t = 1, \ldots, T$ (the value network also predicts the expected return at $t = 0$ based on its own initial states, with the understanding that $\mathbf{a}_{1:0} = \emptyset$ and $\mathbf{r}^\pi_{1:0} = \emptyset$ are empty sets). The value network is trained by minimizing a second objective function

$$\mathcal{L}^v(\phi) = \frac{1}{N_\text{trials}} \sum_{n=1}^{N_\text{trials}} \Big[\mathcal{E}_n(\phi) + \Omega^v_n(\phi) \Big], \tag{4}$$

$$\mathcal{E}_n(\phi) = \frac{1}{T+1} \sum_{t=0}^{T} \left[\sum_{\tau=t}^{T} \rho_{\tau+1} - v_\phi(\mathbf{a}_{1:t}, \mathbf{r}^\pi_{1:t}) \right]^2, \tag{5}$$

where $\Omega^v_n(\phi)$ denotes any regularization terms for the value network. The necessary gradient $\nabla_\phi \mathcal{E}_n(\phi)$ [and likewise $\nabla_\phi \Omega^v_n(\phi)$] is again computed by BPTT.

Policy and value recurrent neural networks

The policy probability distribution $\pi_\theta(\mathbf{a}_t|\mathbf{u}_{1:t})$ and scalar baseline $v_\phi(\mathbf{a}_{1:t}, \mathbf{r}^\pi_{1:t})$ are each represented by an RNN of N firing-rate units $\mathbf{r}^\pi$ and $\mathbf{r}^v$, respectively, where we interpret each unit as the mean firing rate of a group of neurons. In the case where the agent chooses a single action at each time t, the activity of the policy network determines $\pi_\theta(\mathbf{a}_t|\mathbf{u}_{1:t})$ through a linear readout followed by softmax normalization:

$$\mathbf{z}_t = W^\pi_\text{out} \mathbf{r}^\pi_t + \mathbf{b}^\pi_\text{out}, \tag{6}$$

$$\pi_\theta(a_t = k|\mathbf{u}_{1:t}) = \frac{e^{(\mathbf{z}_t)_k}}{\sum_{\ell=1}^{N_a} e^{(\mathbf{z}_t)_\ell}} \tag{7}$$

for $k = 1, \ldots, N_a$. Here W^{π}_{out} is an $N_a \times N$ matrix of connection weights from the units of the policy network to the N_a linear readouts $\mathbf{z}_t$, and $\mathbf{b}^{\pi}_{\text{out}}$ are N_a biases. Action selection is implemented by randomly sampling from the probability distribution in Equation 7, and constitutes an important difference from previous approaches to training RNNs for cognitive tasks, namely, here the final output of the network (during training) is a specific action, not a graded decision variable. From a biological point of view, we consider this sampling as an abstract representation of the downstream action selection mechanisms present in the brain, including the role of noise in implicitly realizing stochastic choices with deterministic outputs (Wang, 2002, 2008). Meanwhile, the activity of the value network predicts future returns through a linear readout

$$v_{\phi}(\mathbf{a}_{1:t}, \mathbf{u}_{1:t}) = W^{v}_{\text{out}} \mathbf{r}^{v}_{t} + b^{v}_{\text{out}}, \tag{8}$$

where W^{v}_{out} is an $1 \times N$ matrix of connection weights from the units of the value network to the single linear readout v_{ϕ}, and b^{v}_{out} is a bias term.

In order to take advantage of recent developments in training RNNs [in particular, addressing the problem of vanishing gradients (Bengio et al., 1994)] while retaining intepretability, we use a modified form of Gated Recurrent Units (GRUs) (Cho et al., 2014; Chung et al., 2014) with a threshold-linear "f-I" curve $[x]_+ = \max(0, x)$ to obtain positive firing rates. These units are thus leaky, threshold-linear units with dynamic time constants and gated recurrent inputs. The equations that describe their dynamics can be derived by a naïve discretization of the following continuous-time equations for the N currents $\mathbf{x}$ and corresponding rectified linear firing rates $\mathbf{r}$:

$$\boldsymbol{\lambda} = \text{sigmoid}(W^{\lambda}_{\text{rec}} \mathbf{r} + W^{\lambda}_{\text{in}} \mathbf{u} + \mathbf{b}^{\lambda}), \tag{9}$$

$$\boldsymbol{\gamma} = \text{sigmoid}(W^{\gamma}_{\text{rec}} \mathbf{r} + W^{\gamma}_{\text{in}} \mathbf{u} + \mathbf{b}^{\gamma}), \tag{10}$$

$$\frac{\tau}{\boldsymbol{\lambda}} \odot \dot{\mathbf{x}} = -\mathbf{x} + W_{\text{rec}}(\boldsymbol{\gamma} \odot \mathbf{r}) + W_{\text{in}} \mathbf{u} + \mathbf{b} + \sqrt{2\tau\sigma^2_{\text{rec}}}\, \boldsymbol{\xi}, \tag{11}$$

$$\mathbf{r} = [\mathbf{x}]_{+}. \tag{12}$$

Here $\dot{\mathbf{x}} = d\mathbf{x}/dt$ is the derivative of $\mathbf{x}$ with respect to time, $\odot$ denotes elementwise multiplication, $\text{sigmoid}(x) = [1 + e^{-x}]^{-1}$ is the logistic sigmoid, $\mathbf{b}^{\lambda}$, $\mathbf{b}^{\gamma}$, and $\mathbf{b}$ are biases, $\boldsymbol{\xi}$ are N independent Gaussian white noise processes with zero mean and unit variance, and σ^2_{rec} controls the size of this noise. The multiplicative gates $\boldsymbol{\lambda}$ dynamically modulate the overall time constant τ for network units, while the $\boldsymbol{\gamma}$ control the recurrent inputs. The $N \times N$ matrices W_{rec}, W^{λ}_{rec}, and W^{γ}_{rec} are the recurrent weight matrices, while the $N \times N_{\text{in}}$ matrices W_{in}, W^{λ}_{in}, and W^{γ}_{in} are connection weights from the N_{in} inputs $\mathbf{u}$ to the N units of the network. We note that in the case where $\boldsymbol{\lambda} \to 1$ and $\boldsymbol{\gamma} \to 1$ the equations reduce to "simple" leaky threshold-linear units without the modulation of the time constants or gating of inputs. We constrain the recurrent connection weights (Song et al., 2016) so that the overall connection probability is p_c; specifically, the number of incoming connections for each unit, or in-degree, was set to exactly $K = p_c N$.

The result of discretizing Equations 9-12, as well as details on initializing the network parameters, are given in Methods. We successfully trained networks with time steps $\Delta t = 1$ ms, but for computational convenience all of the networks in this work were trained and run with $\Delta t = 10$ ms. We note that, for typical tasks in systems neuroscience lasting on the order of several seconds, this already implies trials lasting hundreds of time steps. Unless noted otherwise in the text, all networks were trained using the parameters listed in Table 1.

While the inputs to the policy network π_θ are determined by the environment, the value network always receives as inputs the activity of the policy network $\mathbf{r}^{\pi}$, together with information about which actions were actually selected (Figure 1B). The value network serves two purposes: first, the output of the value network is used as the baseline in the REINFORCE gradient, Equation 3, to reduce the variance of the gradient estimate (Peters & Schaal, 2008); second, since policy gradient reinforcement learning does not explicitly use a value function but value information is nevertheless implicitly contained in the policy, the value network serves as an explicit and potentially nonlinear readout of this information. In situations where expected reward is closely related to confidence, this may explain, for example, certain disassociations between perceptual decisions and reports of the associated confidence (Lak et al., 2014).

A reward baseline, which transforms accumulated rewards into a prediction error (Schultz et al., 1997; Bayer & Glimcher, 2005), is essential to many learning schemes, especially those based on REINFORCE. Indeed, it has been suggested that in general this baseline should be not only task-specific but stimulus (task-condition)-specific (Frémaux et al., 2010; Engel et al., 2015; Miconi, 2016), and that this information may be represented in the orbitofrontal cortex (Wallis, 2007) or basal ganglia (Doya, 2000). Previous schemes, however, did not propose how this baseline "critic" may be instantiated, instead implementing it algorithmically. Here we propose a simple neural implementation of the baseline that automatically depends on the stimulus and thus does not require the learning system to have access to the true trial type, which in general is not known with certainty to the agent.

Tasks with simple input-output mappings

The training procedure described in the previous section can be used for a variety of tasks, and results in networks that qualitatively reproduce both behavioral and electrophysiological findings from experiments with behaving animals. For the example perceptual decision-making task above, the trained network learns to integrate the sensory evidence to make the correct decision about which of two noisy inputs is larger (Figure 1C). This and additional networks trained for the same task were able to reach the target performance in ~7,000 trials starting from completely random connection weights, and moreover the networks learned the "core" task after ~2,000 trials (Figure 1—figure supplement 1). As with monkeys performing the task, longer stimulus durations allow the network to improve its performance by continuing to integrate the incoming sensory evidence (Wang, 2002; Kiani et al., 2008). Indeed, the output of the value network shows that the expected reward (in this case equivalent to confidence) is modulated by stimulus difficulty and increases in time (Figure 1—figure supplement 2); even though a confidence judgment is not strictly required for this task, the baseline value network provides a natural mechanism for extracting information about future reward from the policy network. This gives support to the hypothesis that confidence estimates are a fundamental and ubiquitous component of decision-making (Kepecs et al., 2008).

Sorting the activity of individual units in the network by the signed coherence (the strength of the evidence, with negative values indicating evidence for L and positive for R) also reveals coherence-dependent ramping activity (Figure 1D) as observed in neural recordings from numerous perceptual decision-making experiments, e.g., Roitman & Shadlen (2002). The reaction time as a function of coherence in the reaction-time version of the same task, in which the go cue coincides with the time of stimulus onset, is also shown in Figure 1—figure supplement 3 and may be compared, e.g., to Wang (2002); Mazurek et al. (2003); Wong & Wang (2006). We note that, unlike in many neural models [e.g., Wong & Wang (2006)], here a firing-rate threshold is unnecessary for determining the reaction time; instead, the time at which the network commits to a decision is unambiguously given by the time at which the selected action is L or R. Learning curves for this and additional networks trained for the same reaction-time task are shown in Figure 1—figure supplement 4.

In addition to the example task from the previous section, we trained networks for three well-known behavioral paradigms in which the correct, or optimal, behavior is determined on each trial by the task condition alone. Similar tasks have previously been addressed with several different forms of supervised learning, including FORCE (Sussillo & Abbott, 2009; Carnevale et al., 2015), Hessian-free (Martens & Sutskever, 2011; Mante et al., 2013; Barak et al., 2013), and stochastic gradient descent (Pascanu et al., 2013b; Song et al., 2016), so that the results shown in Figure 2 are presented as confirmation that the same tasks can also be learned using reward feedback on definite actions alone. For all three tasks the pre-stimulus fixation period was 750 ms; the networks had to maintain fixation until the start of a 500-ms "decision" period, which was indicated by the extinction of the fixation cue. At this time the network was required to choose one of two alternatives to indicate its decision and receive a reward of +1 for a correct response and 0 for an incorrect response; otherwise, the networks received a reward of -1.

The context-dependent integration task (Figure 2A) is based on Mante et al. (2013), in which monkeys were required to integrate one type of stimulus (the motion or color of the presented dots) while ignoring the other depending on a context cue. In training the network, we included both the 750-ms stimulus period and 300-1500-ms delay period following stimulus presentation. The delay consisted of 300 ms followed by a variable duration drawn from an exponential distribution with mean 300 ms and truncated at a maximum of 1200 ms. The network successfully learned to perform the task, which is reflected in the psychometric functions showing the percentage of trials on which the network chose R as a function of the signed motion and color coherences, where motion and color indicate the two sources of noisy information and the sign is positive for R and negative for L (Figure 2A, left). As in electrophysiological recordings, units in the policy network show mixed selectivity when sorted by different combinations of task variables (Figure 2A, right). Learning curves for this and additional networks trained for the task are shown in Figure 2—figure supplement 1.

The multisensory integration task (Figure 2B) is based on Raposo et al. (2012, 2014), in which rats used visual flashes and auditory clicks to determine whether the event rate was higher or lower than a learned threshold of 12.5 events per second. When both modalities were presented, they were congruent, which implied that the rats could improve their performance by combining information from both sources. As in the experiment, the network was trained with a 1000-ms stimulus period, with inputs whose magnitudes were proportional (both positively and negatively) to the event rate. For this task the input connection weights W_{in}, W_{in}^{λ}, and W_{in}^{γ} were initialized so that a third of the $N = 150$ policy network units received visual inputs only, another third auditory inputs only, and the remaining third received neither. As shown in the psychometric function (percentage of high choices as a function of event rate, Figure 2B, left), the trained network exhibits multisensory enhancement in which performance on multisensory trials was better than on single-modality trials. Indeed, like rats, the network combined the two modalities "optimally," with

$$\frac{1}{\sigma^2_{\text{visual}}} + \frac{1}{\sigma^2_{\text{auditory}}} \approx \frac{1}{\sigma^2_{\text{multisensory}}}, \tag{13}$$

where σ^2_{visual}, $\sigma^2_{\text{auditory}}$, and $\sigma^2_{\text{multisensory}}$ are the variances obtained from fits of the psychometric functions to cumulative Gaussian functions for visual only, auditory only, and multisensory (both visual and auditory) trials, respectively (Table 2). As observed in electrophysiological recordings, moreover, policy network units exhibit a wide range of tuning to task parameters, with some selective to choice and others to modality, while many units showed mixed selectivity to all task variables (Figure 2B, right). Learning curves for this and additional networks trained for the task are shown in Figure 2—figure supplement 2.

The parametric working memory task (Figure 2C) is based on the vibrotactile frequency discrimination task of Romo et al. (1999), in which monkeys were required to compare the frequencies of two temporally separated stimuli to determine which was higher. For network training, the task epochs consisted of a 500-ms base stimulus with "frequency" f_1, a 2700-3300-ms delay, and a 500-ms comparison stimulus with frequency f_2; for the trials shown in Figure 2C the delay was always 3000 ms as in the experiment. During the decision period, the network had to indicate which stimulus was higher by choosing $f_1 < f_2$ or $f_1 > f_2$. The stimuli were constant inputs with amplitudes proportional (both positively and negatively) to the frequency. For this task we set the learning rate to $\eta = 0.002$; the network successfully learned to perform the task (Figure 2C, left), and the individual units of the network, when sorted by the first stimulus (base frequency) f_1, exhibit highly hetergeneous activity (Figure 2C, right) characteristic of neurons recorded in the prefrontal cortex of monkeys performing the task (Machens et al., 2010). Learning curves for this and additional networks trained for the task are shown in Figure 2—figure supplement 3.

Additional comparisons can be made between the model networks shown in Figure 2 and the neural activity observed in behaving animals, for example state-space analyses as in Mante et al. (2013), Carnevale et al. (2015), or Song et al. (2016). Such comparisons reveal that, as found previously in studies such as Barak et al. (2013), the model networks exhibit many, but not all, features present in electrophysiological recordings. Figure 2 and the following make clear, however, that RNNs trained with reward feedback alone can already reproduce the mixed selectivity characteristic of neural populations in higher cortical areas (Rigotti et al., 2010, 2013), thereby providing a valuable platform for future investigations of how such complex representations are learned.

Confidence and perceptual decision-making

All of the tasks in the previous section have the property that the correct response on any single trial is a function only of the task condition, and, in particular, does not depend on the network's state during the trial. In a postdecision wager task (Kiani & Shadlen, 2009), however, the optimal decision depends on the animal's (agent's) estimate of the probability that its decision is correct, i.e., its confidence. This makes it difficult to train with standard supervised learning: instead, we trained an RNN to perform the task by maximizing overall reward. On a random half of the trials a "sure" option was presented, and selecting this option resulted in a reward that was 0.7 times the size of the reward obtained when correctly choosing L or R. As in the monkey experiment, the network receives no information indicating whether or not a given trial will contain a sure option until the middle of the delay period after stimulus offset, thus ensuring that the network makes a decision about the stimulus on all trials (Figure 3A). For this task the input connection weights W_{in}, W_{in}^{λ}, and W_{in}^{γ} were initialized so that half the units received information about the sure target while the other half received evidence for L and R. All units initially received fixation input.

The key behavioral features found in Kiani & Shadlen (2009); Wei & Wang (2015) are reproduced in the trained network, namely the network opted for the sure option more frequently when the coherence was small or stimulus duration short (Figure 3B, left); and when the network was presented with a sure option but waived it in favor of choosing L or R, the performance was better than on trials when the sure option was not presented (Figure 3B, right). The latter observation is taken as indication that neither monkeys nor trained networks choose the sure target on the basis of stimulus difficulty alone but based on their internal sense of uncertainty on each trial.

Figure 3C shows the activity of an example network unit, sorted by whether the decision was the unit's preferred or nonpreferred target (as determined by firing rates during the stimulus period on all trials), for both non-wager and wager trials. In particular, on trials in which the sure option was chosen, the firing rate is intermediate compared to trials on which the network made a decision by choosing L or R. Learning curves for this and additional networks trained for the task are shown in Figure 3—figure supplement 1.

Value-based economic choice task

We also trained networks to perform the simple economic choice task of Padoa-Schioppa & Assad (2006) and examined the activity of the *value*, rather than policy, network. The choice patterns of the networks were modulated only by varying the reward contingencies (Figure 4A, upper and lower). We note that, on each trial there is a "correct" answer in the sense that there is a choice which results in greater reward. In contrast to the previous tasks, however, information regarding whether an answer is correct in this sense is not contained in the inputs but rather in the association between inputs and rewards. This distinguishes the task from the cognitive tasks discussed in previous sections.

Each trial began with a 750-ms fixation period; the offer, which indicated the "juice" type and amount for the left and right choices, was presented for 1000-2000 ms, followed by a 750-ms decision period during which the network was required to indicate its decision. In the upper panel of Figure 4A the indifference point was set to 1A = 2.2B during training, which resulted in 1A = 2.0B when fit to a cumulative Gaussian (Figure 4—figure supplement 1), while in the lower panel it was set to 1A = 4.1B during training and resulted in 1A = 4.0B (Figure 4—figure supplement 2). The basic unit of reward, i.e., 1B, was 0.1. For this task we increased the initial value of the value network's input weights, W^{v}_{in}, by a factor of 10 to drive the value network more strongly.

Strikingly, the activity of units in the value network v_ϕ exhibits similar types of tuning to task variables as observed in the orbitofrontal cortex of monkeys, with some units (roughly 20% of active units) selective to chosen value, others (roughly 60%, for both A and B) to offer value, and still others (roughly 20%) to choice alone as defined in Padoa-Schioppa & Assad (2006) (Figure 4B). The policy network also contained units with a diversity of tuning. Learning curves for this and additional networks trained for the task are shown in

Figure 4—figure supplement 3. We emphasize that no changes were made to the network architecture for this value-based economic choice task. Instead, the same scheme shown in Figure 1B, in which the value network is responsible for predicting future rewards to guide learning but is *not* involved in the execution of the policy, gave rise to the pattern of neural activity shown in Figure 4B.

Discussion

In this work we have demonstrated reward-based training of recurrent neural networks for both cognitive and value-based tasks. Our main contributions are twofold: first, our work expands the range of tasks and corresponding neural mechanisms that can be studied by analyzing model recurrent neural networks, while providing a unified setting in which to study such diverse computations and compare to electrophysiological recordings from behaving animals; second, by explicitly incorporating reward into network training, our work makes it possible to more directly address the question of reward-dependent processes in the brain, for instance the role of value representation that is essential for learning, but not for executing, a task.

To our knowledge, the specific form of the baseline network inputs used in this work, which was chosen as a biologically plausible candidate architecture for implementing policy search reinforcement learning, has not been used previously in the context of recurrent policy gradients; it combines ideas from Wierstra et al. (2009) where the baseline network received the same inputs as the policy network in addition to the selected actions, and Ranzato et al. (2016), where the baseline was implemented as a simple linear regressor of the activity of the policy network so that the policy and value networks effectively shared the same recurrent units. Indeed, for some of the simpler tasks considered here, models with a baseline consisting of a linear readout of the selected actions and policy network activity could be trained in comparable (but slightly longer) time (Figure 1—figure supplement 5). A separate baseline network, however, suggests an explicit role for value representation in the brain that is essential for learning a task (equivalently, if the environment is changing), but not for executing an already learned task, as is sometimes found in experiments (Turner & Desmurget, 2010; Schoenbaum et al., 2011; Stalnaker et al., 2015). Since an accurate baseline dramatically improves learning but is not *required*—the algorithm takes many more samples to converge with a constant baseline, for instance—this baseline network hypothesis for the role of value representation may account for some of the subtle yet broad learning deficits observed in OFC-lesioned animals (Wallis, 2007). Moreover, since expected reward is closely related to decision confidence in many of the tasks considered, a value network that nonlinearly reads out confidence information from the policy network is consistent with experimental findings in which OFC inactivation affects the ability to report confidence but not decision accuracy (Lak et al., 2014).

Our results thus support the actor-critic picture for reward-based learning, in which one circuit directly computes the policy to be followed, while a second structure, receiving projections from the policy network as well as information about the selected actions, computes expected future reward to guide learning. We note that this is the opposite of the way value functions are usually thought of in neuroscience, which stresses the primacy of the value function from which policies are derived. However, there is experimental evidence that signals in the striatum are more suitable for direct policy search rather than for updating action values as an intermediate step, as would be the case for value function-based approaches to computing the policy (Li & Daw, 2011; Niv & Langdon, 2016). Moreover, although we have used a single RNN each to represent the policy and value modules, using "deep," multilayer RNNs may increase the representational power of each module (Pascanu et al., 2013a). For instance, more complex tasks than considered in this work may require hierarchical feature representation in the policy network, and likewise value networks can use a combination of the different features [including raw sensory inputs (Wierstra et al., 2009)] to predict future reward. Anatomically, the policy networks may correspond to circuits in dorsolateral prefrontal cortex, while the value networks may correspond to circuits in the orbitofrontal cortex (Schultz et al., 2000; Takahashi et al., 2011) or basal ganglia (Hikosaka et al., 2014). This architecture also provides a useful example of the hypothesis that various areas of the brain effectively optimize different cost functions (Marblestone et al., 2016): in this case, the policy network maximizes reward, while the value network minimizes the prediction error for future reward.

As in many other supervised learning approaches used previously to train RNNs (Mante et al., 2013; Song et al., 2016), the use of BPTT to compute the gradients (in particular, the eligibility) make our "plasticity rule" not biologically plausible. Thus our focus has been on learning from realistic feedback signals provided by the environment but not on its physiological implementation. Nevertheless, recent work suggests that exact backpropagation is not necessary (Lillicrap et al., 2014), and approximate forms of backpropagation and SGD can be implemented in a biologically plausible manner (Scellier & Bengio, 2016). Such ideas require further investigation and may lead to effective yet more neurally plausible methods for training model neural networks.

Recently, Miconi (2016) used a "node perturbation"-based (Fiete & Seung, 2006; Fiete et al., 2007; Hoerzer et al., 2014) algorithm with an error signal at the end of each trial to train RNNs for several cognitive tasks, and indeed, node perturbation is closely related to the REINFORCE algorithm used in this work. On one hand, the method described in Miconi (2016) is more biologically plausible in the sense of not requiring gradients computed via backpropagation through time as in our approach; on the other hand, in contrast to the networks in this work, those in Miconi (2016) did not "commit" to a discrete action and thus the error signal was a graded quantity. In this and other works (Frémaux et al., 2010), moreover, the prediction error was computed by algorithmically keeping track of a stimulus (task condition)-specific running average of rewards. Here we propose a simple, concrete scheme for approximating the average that automatically depends on the stimulus, without requiring an external learning system to maintain a separate record for each (true) trial type, which is not known by the agent with certainty.

One of the advantages of the REINFORCE algorithm for policy gradient reinforcement learning is that direct supervised learning can also be mixed with reward-based learning, by including only the eligibility term in Equation 3 without modulating by reward (Mnih et al., 2014), i.e., by maximizing the log-likelihood of the desired actions. Although all of the networks in this work were trained from reward feedback only, it will be interesting to investigate this feature of the REINFORCE algorithm in the context of human learning. Another advantage, which we have not exploited here, is the possibility of learning policies for continuous action spaces (Peters & Schaal, 2008; Wierstra et al., 2009); this would allow us, for example, to model arbitrary saccade targets in the perceptual decision-making task, rather than limiting the network to discrete choices.

We have previously emphasized the importance of incorporating biological constraints in the training of neural networks (Song et al., 2016). For instance, neurons in the mammalian cortex have purely excitatory or inhibitory effects on other neurons, which is a consequence of Dale's Principle for neurotransmitters (Eccles et al., 1954). In this work we did not include such constraints due to the more complex nature of our rectified GRUs (Equations 9-12); in particular, the units we used are capable of dynamically modulating their time constants and gating their recurrent inputs, and we therefore interpreted the firing rate units as a mixture of both excitatory and inhibitory populations. Indeed, these may implement the "reservoir of time constants" observed experimentally (Bernacchia et al., 2011). In the future, however, comparison to both model spiking networks and electrophysiological recordings will be facilitated by including more biological realism, by explicitly separating the roles of excitatory and inhibitory units (Mastrogiuseppe & Ostojic, 2016). Moreover, since both the policy and value networks are obtained by minimizing an objective function, additional regularization terms can be easily included to obtain networks whose activity is more similar to neural recordings (Sussillo et al., 2015; Song et al., 2016).

Finally, one of the most appealing features of RNNs trained to perform many tasks is their ability to provide insights into neural computation in the brain. However, methods for revealing neural mechanisms in such networks remain limited to state-space analysis (Sussillo & Barak, 2013), which in particular does not reveal how the synaptic connectivity leads to the dynamics responsible for implementing the higher-level policy. General and systematic methods for analyzing trained networks are still needed and are the subject of ongoing investigation. Nevertheless, reward-based training of RNNs makes it more likely that the resulting networks will correspond closely to biological networks observed in experiments with behaving animals. We expect that the continuing development of tools for training model neural networks in neuroscience will thus contribute novel insights into the neural basis of animal cognition.

Methods

Policy gradient reinforcement learning with RNNs

Here we review the application of the REINFORCE algorithm for policy gradient reinforcement learning to recurrent neural networks (Williams, 1992; Sutton & Barto, 1998; Peters & Schaal, 2008; Wierstra et al., 2009). In particular, we provide a careful derivation of Equation 3 following, in part, the exposition in Zaremba & Sutskever (2016).

Let $H_{\mu:t}$ be the sequence of interactions between the environment and agent (i.e., the environmental states, observables, and agent actions) that results in the environment being in state $\mathbf{s}_{t+1}$ at time $t+1$ starting from state $\mathbf{s}_\mu$ at time μ:

$$H_{\mu:t} = (\mathbf{s}_{\mu+1:t+1}, \mathbf{u}_{\mu:t}, \mathbf{a}_{\mu:t}). \tag{14}$$

For notational convenience in the following, we adopt the convention that, for the special case of $\mu = 0$, the history $H_{0:t}$ includes the initial state $\mathbf{s}_0$ and excludes the meaningless inputs $\mathbf{u}_0$, which are not seen by the agent:

$$H_{0:t} = (\mathbf{s}_{0:t+1}, \mathbf{u}_{1:t}, \mathbf{a}_{0:t}). \tag{15}$$

When $t = 0$, it is also understood that $\mathbf{u}_{1:0} = \emptyset$, the empty set. A full history, or a trial, is thus denoted as

$$H \equiv H_{0:T} = (\mathbf{s}_{0:T+1},\ \mathbf{u}_{1:T},\ \mathbf{a}_{0:T}), \tag{16}$$

where T is the end of the trial. Indeed, since different trials can have different durations, we take T to be the maximum length of a trial in the task. The reward ρ_{t+1} at time $t+1$ following actions $\mathbf{a}_t$ (we use ρ to distinguish it from the firing rates $\mathbf{r}$ of the RNNs) is determined by this history, which we sometimes indicate explicitly by writing

$$\rho_{t+1} = \rho_{t+1}(H_{0:t}). \tag{17}$$

As noted in the main text, we adopt the convention that the agent performs actions at $t = 0,\ldots,T$ and receives rewards at $t = 1,\ldots,T+1$ to emphasize that rewards follow the actions and are jointly determined with the next state (Sutton & Barto, 1998). For notational simplicity, here and elsewhere we assume that any discount factor is already included in ρ_{t+1}, i.e., in all places where the reward appears we consider $\rho_{t+1} \to e^{-t/\tau_{\text{reward}}}\rho_{t+1}$, where τ_{reward} is the time constant for discounting future rewards (Doya, 2000); we included temporal discounting only for the reaction-time version of the simple perceptual decision-making task (Figure 1—figure supplement 3), where we set $\tau_{\text{reward}} = 10$ s. For the remaining tasks, $\tau_{\text{reward}} = \infty$.

Explicitly, a trial $H_{0:T}$ comprises the following. At time $t = 0$, the environment is in state $\mathbf{s}_0$ with probability $\mathcal{E}(\mathbf{s}_0)$. The agent initially chooses a set of actions $\mathbf{a}_0$ with probability $\pi_\theta(\mathbf{a}_0)$, which is determined by the parameters of the policy network, in particular the initial conditions $\mathbf{x}_0$ and readout weights W^{π}_{out} and biases $\mathbf{b}^{\pi}_{\text{out}}$ (Equation 6). At time $t = 1$, the environment, depending on its previous state $\mathbf{s}_0$ and the agent's actions $\mathbf{a}_0$, transitions to state $\mathbf{s}_1$ with probability $\mathcal{E}(\mathbf{s}_1|\mathbf{s}_0,\mathbf{a}_0)$. The history up to this point is $H_{0:0} = (\mathbf{s}_{0:1}, \emptyset, \mathbf{a}_{0:0})$, where $\emptyset$ indicates that no inputs have yet been seen by the network. The environment also generates reward ρ_1, which depends on this history, $\rho_1 = \rho_1(H_{0:0})$. From state $\mathbf{s}_1$ the environment generates observables (inputs to the agent) $\mathbf{u}_1$ with a distribution given by $\mathcal{E}(\mathbf{u}_1|\mathbf{s}_1)$. In response, the agent, depending on the inputs $\mathbf{u}_1$ it receives from the environment, chooses the set of actions $\mathbf{a}_1$ according to the distribution

$\pi_\theta(\mathbf{a}_1|\mathbf{u}_{1:1}) = \pi_\theta(\mathbf{a}_1|\mathbf{u}_1)$. The environment, depending on its previous states $\mathbf{s}_{0:1}$ and the agent's previous actions $\mathbf{a}_{0:1}$, then transitions to state $\mathbf{s}_2$ with probability $\mathscr{E}(\mathbf{s}_2|\mathbf{s}_{0:1},\mathbf{a}_{0:1})$. Thus $H_{0:1} = (\mathbf{s}_{0:2},\mathbf{u}_{1:1},\mathbf{a}_{0:1})$. Iterating these steps, the history at time t is therefore given by Equation 15, while a full history is given by Equation 16.

The probability $p_\theta(H_{0:\tau})$ of a particular sub-history $H_{0:\tau}$ up to time τ occurring, under the policy π_θ parametrized by θ, is given by

$$p_\theta(H_{0:\tau}) = \left[\prod_{t=1}^{\tau} \mathscr{E}(\mathbf{s}_{t+1}|\mathbf{s}_{0:t},\mathbf{a}_{0:t})\pi_\theta(\mathbf{a}_t|\mathbf{u}_{1:t})\mathscr{E}(\mathbf{u}_t|\mathbf{s}_t)\right]\mathscr{E}(\mathbf{s}_1|\mathbf{s}_0,\mathbf{a}_0)\pi_\theta(\mathbf{a}_0)\mathscr{E}(\mathbf{s}_0). \tag{18}$$

In particular, the probability $p_\theta(H)$ of a history $H = H_{0:T}$ occurring is

$$p_\theta(H) = \left[\prod_{t=1}^{T} \mathscr{E}(\mathbf{s}_{t+1}|\mathbf{s}_{0:t},\mathbf{a}_{0:t})\pi_\theta(\mathbf{a}_t|\mathbf{u}_{1:t})\mathscr{E}(\mathbf{u}_t|\mathbf{s}_t)\right]\mathscr{E}(\mathbf{s}_1|\mathbf{s}_0,\mathbf{a}_0)\pi_\theta(\mathbf{a}_0)\mathscr{E}(\mathbf{s}_0). \tag{19}$$

A key ingredient of the REINFORCE algorithm is that the logarithmic derivatives of Equation 18 with respect to the parameters θ do not depend on the unknown (to the agent) environmental dynamics contained in $\mathscr{E}$, because

$$\nabla_\theta \log p_\theta(H_{0:\tau}) = \sum_{t=0}^{\tau} \nabla_\theta \log \pi_\theta(\mathbf{a}_t|\mathbf{u}_{1:t}), \tag{20}$$

with the understanding that $\mathbf{u}_{1:0} = \emptyset$ (the empty set) and therefore $\pi_\theta(\mathbf{a}_0|\mathbf{u}_{1:0}) = \pi_\theta(\mathbf{a}_0)$.

The goal of the agent is to maximize the expected return at time $t = 0$ (Equation 1, reproduced here)

$$J(\theta) = \mathbb{E}_H\left[\sum_{\tau=0}^{T} \rho_{\tau+1}(H_{0:\tau})\right], \tag{21}$$

where we have used the time index τ for notational consistency with the following and made the history-dependence of the rewards explicit. In terms of the probability of each history H occurring, Equation 19, we have

$$J(\theta) = \sum_H p_\theta(H)\left[\sum_{\tau=0}^{T} \rho_{\tau+1}(H_{0:\tau})\right], \tag{22}$$

where the generic sum over H may include both sums over discrete variables and integrals over continuous variables. Using the "likelihood trick"

$$\nabla_\theta f(\theta) = f(\theta)\frac{\nabla_\theta f(\theta)}{f(\theta)} = f(\theta)\nabla_\theta \log f(\theta), \tag{23}$$

we can write

$$\begin{aligned} \nabla_\theta J(\theta) &= \sum_H \left[\nabla_\theta p_\theta(H)\right] \left[\sum_{\tau=0}^{T} \rho_{\tau+1}(H_{0:\tau})\right] && (24) \\ &= \sum_H p_\theta(H) \left[\nabla_\theta \log p_\theta(H)\right] \left[\sum_{\tau=0}^{T} \rho_{\tau+1}(H_{0:\tau})\right] && (25) \\ &= \sum_{\tau=0}^{T} \sum_H p_\theta(H) \left[\nabla_\theta \log p_\theta(H)\right] \rho_{\tau+1}(H_{0:\tau}). && (26) \end{aligned}$$

Since, for any $\tau = 0, \ldots, T$,

$$p_\theta(H) = p_\theta(H_{0:T}) = p_\theta(H_{\tau+1:T}|H_{0:\tau})p_\theta(H_{0:\tau}) \tag{27}$$

(cf. Equation 18), we can take the logarithm and insert into Equation 26 to obtain

$$\begin{aligned} \nabla_\theta J(\theta) = &\sum_{\tau=0}^{T} \sum_H p_\theta(H) \Big[\nabla_\theta \log p_\theta(H_{0:\tau})\Big] \rho_{\tau+1}(H_{0:\tau}) \\ &+ \sum_{\tau=0}^{T} \sum_H p_\theta(H) \Big[\nabla_\theta \log p_\theta(H_{\tau+1:T}|H_{0:\tau})\Big] \rho_{\tau+1}(H_{0:\tau}), && (28) \end{aligned}$$

where the second term is zero because (again using Equation 27)

$$\begin{aligned}
&\sum_{\tau=0}^{T}\sum_{H} p_\theta(H)\Big[\nabla_\theta \log p_\theta(H_{\tau+1:T}|H_{0:\tau})\Big]\rho_{\tau+1}(H_{0:\tau}) \\
&\quad= \sum_{\tau=0}^{T}\sum_{H_{0:\tau}} p_\theta(H_{0:\tau})\rho_{\tau+1}(H_{0:\tau}) \sum_{H_{\tau+1:T}} p_\theta(H_{\tau+1:T}|H_{0:\tau})\Big[\nabla_\theta \log p_\theta(H_{\tau+1:T}|H_{0:\tau})\Big] && (29)\\
&\quad= \sum_{\tau=0}^{T}\sum_{H_{0:\tau}} p_\theta(H_{0:\tau})\rho_{\tau+1}(H_{0:\tau})\nabla_\theta \sum_{H_{\tau+1:T}} p_\theta(H_{\tau+1:T}|H_{0:\tau}) && (30)\\
&\quad= \sum_{\tau=0}^{T}\sum_{H_{0:\tau}} p_\theta(H_{0:\tau})\rho_{\tau+1}(H_{0:\tau})\nabla_\theta 1 && (31)\\
&\quad= 0. && (32)
\end{aligned}$$

Therefore, from Equation 20 and the remaining term in Equation 28, we obtain

$$\begin{aligned}
\nabla_\theta J(\theta) &= \sum_{\tau=0}^{T}\sum_{H} p_\theta(H)\left[\sum_{t=0}^{\tau}\nabla_\theta \log \pi_\theta(\mathbf{a}_t|\mathbf{u}_{1:t})\right]\rho_{\tau+1}(H_{0:\tau}) && (33)\\
&= \sum_{H} p_\theta(H)\sum_{\tau=0}^{T}\sum_{t=0}^{\tau}\nabla_\theta \log \pi_\theta(\mathbf{a}_t|\mathbf{u}_{1:t})\rho_{\tau+1}(H_{0:\tau}) && (34)\\
&= \sum_{H} p_\theta(H)\sum_{t=0}^{T}\sum_{\tau=t}^{T}\nabla_\theta \log \pi_\theta(\mathbf{a}_t|\mathbf{u}_{1:t})\rho_{\tau+1}(H_{0:\tau}) && (35)\\
&= \sum_{H} p_\theta(H)\sum_{t=0}^{T}\nabla_\theta \log \pi_\theta(\mathbf{a}_t|\mathbf{u}_{1:t})\left[\sum_{\tau=t}^{T}\rho_{\tau+1}(H_{0:\tau})\right]. && (36)
\end{aligned}$$

We then obtain the first terms of Equation 2 and Equation 3 by estimating the sum over all H by N_{trials} samples from the agent's experience.

In Equation 22 it is evident that subtracting a constant baseline for each parameter will not affect the *gradient* of $J(\theta)$, i.e., Equation 36. It is possible to use this invariance to find an "optimal" value of the constant baseline that minimizes the variance of the gradient estimate (Peters & Schaal, 2008). In practice, however, it is more useful to have a history-dependent baseline that attempts to predict the future return at every time (Wierstra et al., 2009; Zaremba & Sutskever, 2016). We therefore introduce a second network, called the *value network*, that uses the selected actions $\mathbf{a}_{1:t}$ and the activity of the policy network $\mathbf{r}^{\pi}_{1:t}$ to predict the future return $\sum_{\tau=t}^{T} \rho_{\tau+1}$ by minimizing the squared error (Equations 4-5). Intuitively, such a baseline is appealing because the terms in the gradient of Equation 3 are nonzero only if the actual return deviates from what was predicted by the value network.

Discretized network equations and initialization

Carrying out the discretization of Equations 9-12 in time steps of Δt, we obtain

$$\boldsymbol{\lambda}_t = \text{sigmoid}(W_{\text{rec}}^{\lambda}\mathbf{r}_{t-1} + W_{\text{in}}^{\lambda}\mathbf{u}_t + \mathbf{b}^{\lambda}), \tag{37}$$

$$\boldsymbol{\gamma}_t = \text{sigmoid}(W_{\text{rec}}^{\gamma}\mathbf{r}_{t-1} + W_{\text{in}}^{\gamma}\mathbf{u}_t + \mathbf{b}^{\gamma}), \tag{38}$$

$$\mathbf{x}_t = (1-\alpha\boldsymbol{\lambda}_t)\odot\mathbf{x}_{t-1} + \alpha\boldsymbol{\lambda}_t\odot\left[W_{\text{rec}}(\boldsymbol{\gamma}_t\odot\mathbf{r}_{t-1}) + W_{\text{in}}\mathbf{u}_t + \mathbf{b} + \sqrt{2\alpha^{-1}\sigma_{\text{rec}}^2}\,\mathbf{N}(0,1)\right], \tag{39}$$

$$\mathbf{r}_t = [\mathbf{x}_t]_+ \tag{40}$$

for $t = 1,\ldots,T$, where $\alpha = \Delta t/\tau$ and $\mathbf{N}(0,1)$ are normally distributed random numbers with zero mean and unit variance.

The biases $\mathbf{b}^{\lambda}$, $\mathbf{b}^{\gamma}$, and $\mathbf{b}$, as well as the readout weights W_{out}^{π} and W_{out}^{v}, were initialized to zero. The biases for the policy readout $\mathbf{b}_{\text{out}}^{\pi}$ were initially set to zero, while the value network bias b_{out}^{v} was initially set to the “reward” for an aborted trial, -1. The entries of the input weight matrices W_{in}^{γ}, W_{in}^{λ}, and W_{in} for both policy and value networks were drawn from a zero-mean Gaussian distribution with variance K/N_{in}^2. For the recurrent weight matrices W_{rec}, W_{rec}^{λ}, and W_{rec}^{γ}, the K nonzero entries in each row were initialized from a gamma distribution $\Gamma(\alpha,\beta)$ with $\alpha = \beta = 4$, with each entry multiplied randomly by ± 1; the entire matrix was then scaled such that the spectral radius—the largest absolute value of the eigenvalues—was exactly ρ_0. Although we also successfully trained networks starting from normally distributed weights, we found it convenient to control the sign and magnitude of the weights independently. The initial conditions $\mathbf{x}_0$, which are also trained, were set to 0.5 for all units before the start of training. We implemented the networks in the Python machine learning library Theano (The Theano Development Team, 2016).

Adam SGD with gradient clipping

We use a recently developed version of stochastic gradient descent known as Adam, for *ada*ptive *m*oment estimation (Kingma & Ba, 2015), together with gradient clipping to prevent exploding gradients (Graves, 2013; Pascanu et al., 2013b). For clarity, in this section we use vector notation $\boldsymbol{\theta}$ to indicate the set of all parameters being optimized and the subscript k to indicate a specific parameter θ_k. At each iteration $i > 0$, let

$$\mathbf{g}^{(i)} = \frac{\partial \mathscr{L}}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(i-1)}} \tag{41}$$

be the gradient of the objective function $\mathscr{L}$ with respect to the parameters $\boldsymbol{\theta}$. We first clip the gradient if its norm $|\mathbf{g}^{(i)}|$ exceeds a maximum Γ (see Table 1), i.e.,

$$\hat{\mathbf{g}}^{(i)} = \mathbf{g}^{(i)} \times \min\left(1, \frac{\Gamma}{|\mathbf{g}^{(i)}|}\right). \tag{42}$$

Each parameter θ_k is then updated according to

$$\theta_k^{(i)} = \theta_k^{(i-1)} - \eta \frac{\sqrt{1-\beta_2^i}}{1-\beta_1^i} \frac{m_k^{(i)}}{\sqrt{v_k^{(i)}}+\epsilon}, \tag{43}$$

where η is the base learning rate and the moving averages

$$\mathbf{m}^{(i)} = \beta_1 \mathbf{m}^{(i-1)} + (1-\beta_1)\hat{\mathbf{g}}^{(i)}, \tag{44}$$

$$\mathbf{v}^{(i)} = \beta_2 \mathbf{v}^{(i-1)} + (1-\beta_2)\big[\hat{\mathbf{g}}^{(i)}\big]^2 \tag{45}$$

estimate the first and second (uncentered) moments of the gradient. Initially, $\mathbf{m}^{(0)} = \mathbf{v}^{(0)} = 0$. These moments allow each parameter to be updated in Equation 43 according to adaptive learning rates, such that parameters whose gradients exhibit high uncertainty and hence large "signal-to-noise ratio" lead to smaller learning rates.

Except for the base learning rate η (see Table 1), we used the parameter values suggested in Kingma & Ba (2015):

$$\begin{aligned}\beta_1 &= 0.9,\\ \beta_2 &= 0.999,\\ \varepsilon &= 10^{-8}.\end{aligned}$$

Computer code

All code used in this work, including code for generating the figures, is available at `http://github.com/xjwanglab/pyrl`.

Acknowledgements

We thank R. Pascanu, K. Cho, E. Ohran, and E. Russek for valuable discussions. This work was supported by Office of Naval Research Grant N00014-13-1-0297 and a Google Computational Neuroscience Grant.

Competing interests

The authors declare that no competing interests exist.

Author contributions

HFS, participated in all aspects of the project; GRY, analysis and interpretation of data, drafting and revising the article; X-JW, conception and design, analysis and interpretation of data, drafting and revising the article.

References

Barak, O., Sussillo, D., Romo, R., Tsodyks, M., & Abbott, L. F. (2013). From fixed points to chaos: Three models of delayed discrimination. Prog. Neurobiol. 103, 214–22. http://dx.doi.org/10.1016/j.pneurobio.2013.02.002.

Barto, A. G., Sutton, R. S., & Anderson, C. W. (1983). Neuronlike adaptive elements that can solve difficult learning control problems. IEEE Trans. Syst. Man Cybern. SMC-13, 834–846. http://dx.doi.org/10.1109/TSMC.1983.6313077.

Bayer, H. M. & Glimcher, P. W. (2005). Midbrain dopamine neurons encode a quantitative reward prediction error signal. Neuron 47, 129–141. http://dx.doi.org/10.1016/j.neuron.2005.05.020.

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Networks 5, 157–166. http://dx.doi.org/10.1109/72.279181.

Bernacchia, A., Seo, H., Lee, D., & Wang, X.-J. (2011). A reservoir of time constants for memory traces in cortical neurons. Nat. Neurosci. 14, 366–372. http://dx.doi.org/10.1038/nn.2752.

Brosch, T., Neumann, H., & Roelfsema, P. R. (2015). Reinforcement learning of linking and tracing contours in recurrent neural networks. PLOS Comp. Bio. 11, e1004489. http://dx.doi.org/10.1371/journal.pcbi.1004489.

Cadieu, C. F., Hong, H., Yamins, D. L. K., Pinto, N., Ardila, D., Solomon, E. A., Majaj, N. J., & DiCarlo, J. J. (2014). Deep neural networks rival the representation of primate IT cortex for core visual object recognition. PLOS Comp. Bio. 10, e1003963. http://dx.doi.org/10.1371/journal.pcbi.1003963.

Carnevale, F., de Lafuente, V., Romo, R., Barak, O., & Parga, N. (2015). Dynamic control of response criterion in premotor cortex during perceptual detection under temporal uncertainty. Neuron 86, 1067–1077. http://dx.doi.org/10.1016/j.neuron.2015.04.014.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv:1406.1078. http://arxiv.org/abs/1406.1078.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555. http://arxiv.org/abs/1412.3555.

Dayan, P. & Daw, N. D. (2008). Decision theory, reinforcement learning, and the brain. Cogn. Affect. Behav. Neurosci. 8, 429–453. http://dx.doi.org/10.3758/CABN.8.4.429.

Doya, K. (2000). Reinforcement learning in continuous time and space. Neural Comput. 12, 219–245. http://dx.doi.org/10.1162/089976600300015961.

Eccles, J. C., Fatt, P., & Koketsu, K. (1954). Cholinergic and inhibitory synapses in a pathway from motor-axon collaterals to motoneurones. J. Physiol. 126, 524–562. http://dx.doi.org/10.1113/jphysiol.1954.sp005226.

Engel, T. A., Chaisangmongkon, W., Freedman, D. J., & Wang, X.-J. (2015). Choice-correlated activity fluctuations underlie learning of neuronal category representation. Nat. Commun. 6, 6454. http://dx.doi.org/10.1038/ncomms7454.

Fiete, I. R., Fee, M. S., & Seung, H. S. (2007). Model of birdsong learning based on gradient estimation by dynamic perturbation of neural conductances. J. Neurophysiol. 98, 2038–2057. http://dx.doi.org/10.1152/jn.01311.2006.

Fiete, I. R. & Seung, H. S. (2006). Gradient learning in spiking neural networks by dynamic perturbation of conductances. Phys. Rev. Lett. 97, 048104. http://dx.doi.org/10.1103/PhysRevLett.97.048104.

Frank, M. J. & Claus, E. D. (2006). Anatomy of a decision: striato-orbitofrontal interactions in reinforcement learning, decision making, and reversal. Psychol. Rev. 113, 300–326. http://dx.doi.org/10.1037/0033-295X.113.2.300.

Frémaux, N., Sprekeler, H., & Gerstner, W. (2010). Functional requirements for reward-modulated spike-timing-dependent plasticity. J. Neurosci. 30, 13326–13337. http://dx.doi.org/10.1523/JNEUROSCI.6249-09.2010.

Friedrich, J. & Lengyel, M. (2016). Goal-directed decision making with spiking neurons. J. Neurosci. 36, 1529–1546. http://dx.doi.org/10.1523/JNEUROSCI.2854-15.2016.

Gao, P. & Ganguli, S. (2015). On simplicity and complexity in the brave new world of large-scale neuroscience. Curr. Opin. Neurobiol. 32, 148–155.

http://dx.doi.org/10.1016/j.conb.2015.04.003.

Gold, J. I. & Shadlen, M. N. (2007). The neural basis of decision making. Annu. Rev. Neurosci. 30, 535–574. http://dx.doi.org/10.1146/annurev.neuro.29.051605.113038.

Graves, A. (2013). Generating sequences with recurrent neural networks. arXiv:1308.0850. http://arxiv.org/abs/1308.0850.

Grondman, I., Busoniu, L., Lopes, G. A. D., & Babuska, R. (2012). A survey of actor-critic reinforcement learning: Standard and natural policy gradients. IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. 42, 1291–1307.

Hennequin, G., Vogels, T. P., & Gerstner, W. (2014). Optimal control of transient dynamics in balanced networks supports generation of complex movements. Neuron 82, 1394–1406. http://dx.doi.org/10.1016/j.neuron.2014.04.045.

Hikosaka, O., Kim, H. F., Yasuda, M., & Yamamoto, S. (2014). Basal ganglia circuits for reward value-guided behavior. Annu. Rev. Neurosci. 37, 289–306. http://dx.doi.org/10.1146/annurev-neuro-071013-013924.

Hoerzer, G. M., Legenstein, R., & Maass, W. (2014). Emergence of complex computational structures from chaotic neural networks through reward-modulated hebbian learning. Cereb. Cortex 24, 677–690. http://dx.doi.org/10.1093/cercor/bhs348.

Hong, H., Yamins, D. L. K., Majaj, N. J., & DiCarlo, J. J. (2016). Explicit information for category-orthogonal object properties increases along the ventral stream. Nat. Neurosci. 19. http://dx.doi.org/10.1038/nn.4247.

Izhikevich, E. M. (2007). Solving the distal reward problem through linkage of STDP and dopamine signaling. Cereb. Cortex 17, 2443–2452. http://dx.doi.org/10.1093/cercor/bhl152.

Kaelbling, L. P., Littman, M. L., & Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. Artif. Intell. 101, 99–134. http://dx.doi.org/10.1016/S0004-3702(98)00023-X.

Kepecs, A., Uchida, N., Zariwala, H., & Mainen, Z. F. (2008). Neural correlates, computation and behavioural impact of decision confidence. Nature 455, 227–231.

http://dx.doi.org/10.1038/nature07200.

Kiani, R., Hanks, T. D., & Shadlen, M. N. (2008). Bounded integration in parietal cortex underlies decisions even when viewing duration is dictated by the environment. J. Neurosci. 28, 3017–3029. http://dx.doi.org/10.1523/JNEUROSCI.4761-07.2008.

Kiani, R. & Shadlen, M. N. (2009). Representation of confidence associated with a decision by neurons in the parietal cortex. Science 324, 759–64. http://dx.doi.org/10.1126/science.1169405.

Kingma, D. P. & Ba, J. L. (2015). Adam: A method for stochastic optimization. Int. Conf. Learn. Represent. https://arxiv.org/abs/1412.6980.

Laje, R. & Buonomano, D. V. (2013). Robust timing and motor patterns by taming chaos in recurrent neural networks. Nat. Neurosci. 16, 925–33. http://dx.doi.org/10.1038/nn.3405.

Lak, A., Costa, G. M., Romberg, E., Koulakov, A. A., Mainen, Z. F., & Kepecs, A. (2014). Orbitofrontal cortex is required for optimal waiting based on decision confidence. Neuron 84, 190–201. http://dx.doi.org/10.1016/j.neuron.2014.08.039.

Li, J. & Daw, N. D. (2011). Signals in human striatum are appropriate for policy update rather than value prediction. J. Neurosci. 31, 5504–5511. http://dx.doi.org/10.1523/JNEUROSCI.6316-10.2011.

Lillicrap, T. P., Cownden, D., Tweed, D. B., & Akerman, C. J. (2014). Random feedback weights support learning in deep neural networks. arXiv:1411.0247. http://www.arxiv.org/pdf/1411.0247.pdf.

Machens, C. K., Romo, R., & Brody, C. D. (2010). Functional, but not anatomical, separation of "what" and "when" in prefrontal cortex. J. Neurosci. 30, 350–360. http://dx.doi.org/10.1523/JNEUROSCI.3276-09.2010.

Mante, V., Sussillo, D., Shenoy, K. V., & Newsome, W. T. (2013). Context-dependent computation by recurrent dynamics in prefrontal cortex. Nature 503, 78–84. http://dx.doi.org/10.1038/nature12742.

Marblestone, A. H., Wayne, G., & Kording, K. P. (2016). Towards an integration of deep learning and neuroscience. bioRxiv. http://dx.doi.org/10.1101/058545.

Martens, J. & Sutskever, I. (2011). Learning recurrent neural networks with Hessian-free optimization. Proc. 28th Int. Conf. Mach. Learn. http://www.icml-2011.org/papers/532_icmlpaper.pdf.

Mastrogiuseppe, F. & Ostojic, S. (2016). Intrinsically-generated fluctuating activity in excitatory-inhibitory networks. arXiv:1605.04221. http://arxiv.org/abs/1605.04221.

Mazurek, M. E., Roitman, J. D., Ditterich, J., & Shadlen, M. N. (2003). A role for neural integrators in perceptual decision making. Cereb. Cortex 13, 1257–1269. http://dx.doi.org/10.1093/cercor/bhg097.

Miconi, T. (2016). Biologically plausible learning in recurrent neural networks for flexible decision tasks. bioRxiv. http://dx.doi.org/10.1101/057729.

Mnih, V., Hess, N., Graves, A., & Kavukcuoglu, K. (2014). Recurrent models of visual attention. Adv. Neural Inf. Process. Syst. https://papers.nips.cc/paper/5542-recurrent-models-of-visual-attention.pdf.

Niv, Y. & Langdon, A. (2016). Reinforcement learning with Marr. Curr. Opin. Behav. Sci. 11, 67–73. http://dx.doi.org/10.1016/j.cobeha.2016.04.005.

Padoa-Schioppa, C. & Assad, J. A. (2006). Neurons in the orbitofrontal cortex encode economic value. Nature 441, 223–226. http://dx.doi.org/10.1038/nature04676.

Pascanu, R., Gülçehre, Ç., Cho, K., Bengio, Y., Gulcehre, C., Cho, K., & Bengio, Y. (2013a). How to construct deep recurrent neural networks. Int. Conf. Learn. Represent. http://arxiv.org/abs/1312.6026.

Pascanu, R., Mikolov, T., & Bengio, Y. (2013b). On the difficulty of training recurrent neural networks. Proc. 30th Int. Conf. Mach. Learn. http://jmlr.org/proceedings/papers/v28/pascanu13.pdf.

Peters, J. & Schaal, S. (2008). Reinforcement learning of motor skills with policy gradients. Neural Networks 21, 682–697.

Rajan, K., Harvey, C. D., & Tank, D. W. (2015). Recurrent network models of sequence generation and memory. Neuron 90, 128–142. http://dx.doi.org/10.1016/j.neuron.2016.02.009.

Ranzato, M., Chopra, S., Auli, M., & Zaremba, W. (2016). Sequence level training with recurrent neural networks. ICLR pp. 1–15. http://arxiv.org/abs/1511.06732.

Rao, R. P. N. (2010). Decision making under uncertainty: A neural model based on partially observable Markov decision processes. Front. Comput. Neurosci. 4, 146. http://dx.doi.org/10.3389/fncom.2010.00146.

Raposo, D., Kaufman, M. T., & Churchland, A. K. (2014). A category-free neural population supports evolving demands during decision-making. Nat. Neurosci. 17, 1784–1792. http://dx.doi.org/10.1038/nn.3865.

Raposo, D., Sheppard, J. P., Schrater, P. R., & Churchland, A. K. (2012). Multisensory decision-making in rats and humans. J. Neurosci. 32, 3726–35. http://dx.doi.org/10.1523/JNEUROSCI.4998-11.2012.

Rigotti, M., Barak, O., Warden, M. R., Wang, X.-J., Daw, N. D., Miller, E. K., & Fusi, S. (2013). The importance of mixed selectivity in complex cognitive tasks. Nature 497, 585–590. http://dx.doi.org/10.1038/nature12160.

Rigotti, M., Rubin, D. B. D., Wang, X.-J., & Fusi, S. (2010). Internal representation of task rules by recurrent dynamics: the importance of the diversity of neural responses. Front. Comput. Neurosci. 4, 24. http://dx.doi.org/10.3389/fncom.2010.00024.

Roitman, J. D. & Shadlen, M. N. (2002). Response of neurons in the lateral intraparietal area during a combined visual discrimination reaction time task. J. Neurosci. 22, 9475–89. http://www.jneurosci.org/content/22/21/9475.long.

Romo, R., Brody, C. D., Hernández, A., & Lemus, L. (1999). Neuronal correlates of parametric working memory in the prefrontal cortex. Nature 399, 470–473. http://dx.doi.org/10.1038/20939.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In Parallel Distributed Processing, Vol. 1, D. E. Rumelhart & J. L. McClelland, eds. (Cambridge, MA: MIT Press) chap. 8, pp. 318–362.

Scellier, B. & Bengio, Y. (2016). Towards a biologically plausible backprop. arXiv:1602.05179. http://arxiv.org/abs/1602.05179.

Schoenbaum, G., Takahashi, Y., Liu, T. L., & Mcdannald, M. A. (2011). Does the orbitofrontal cortex signal value? Ann. N. Y. Acad. Sci. 1239, 87–99. http://dx.doi.org/10.1111/j.1749-6632.2011.06210.x.

Schultz, W., Dayan, P., & Montague, P. R. (1997). A neural substrate of prediction and reward. Science 275, 1593–1599.

Schultz, W., Tremblay, L., & Hollerman, J. R. (2000). Reward processing in primate orbitofrontal cortex and basal ganglia. Cereb. Cortex 10, 272–84. http://dx.doi.org/10.1093/cercor/10.3.272.

Seung, S. (2003). Learning in spiking neural networks by reinforcement of stochastics transmission. Neuron 40, 1063–1073. http://dx.doi.org/10.1016/S0896-6273(03)00761-X.

Soltani, A., Lee, D., & Wang, X. J. (2006). Neural mechanism for stochastic behaviour during a competitive game. Neural Networks 19, 1075–1090. http://dx.doi.org/10.1016/j.neunet.2006.05.044.

Soltani, A. & Wang, X. J. (2010). Synaptic computation underlying probabilistic inference. Nat Neurosci 13, 112–119. http://dx.doi.org/10.1038/nn.2450.

Song, H. F., Yang, G. R., & Wang, X.-J. (2016). Training excitatory-inhibitory recurrent neural networks for cognitive tasks: A simple and flexible framework. PLOS Comp. Bio. 12, e1004792. http://dx.doi.org/10.1371/journal.pcbi.1004792.

Stalnaker, T. A., Cooch, N. K., & Schoenbaum, G. (2015). What the orbitofrontal cortex does not do. Nat. Neurosci. 18, 620–627. http://dx.doi.org/10.1038/nn.3982.

Sugrue, L. P., Corrado, G. S., & Newsome, W. T. (2005). Choosing the greater of two goods: neural currencies for valuation and decision making. Nat. Rev. Neurosci. 6, 363–75. http://www.ncbi.nlm.nih.gov/pubmed/15832198.

Sussillo, D. (2014). Neural circuits as computational dynamical systems. Curr. Opin. Neurobiol. 25, 156–163. http://dx.doi.org/10.1016/j.conb.2014.01.008.

Sussillo, D. & Abbott, L. F. (2009). Generating coherent patterns of activity from chaotic neural networks. Neuron 63, 544–57. http://dx.doi.org/10.1016/j.neuron.2009.07.018.

Sussillo, D. & Barak, O. (2013). Opening the black box: Low-dimensional dynamics in high-dimensional recurrent neural networks. Neural Comput. 25, 626–49. http://dx.doi.org/10.1162/NECO_a_00409.

Sussillo, D., Churchland, M. M., Kaufman, M. T., & Shenoy, K. V. (2015). A neural network that finds a naturalistic solution for the production of muscle activity. Nat. Neurosci. 18, 1025–1033. http://dx.doi.org/10.1038/nn.4042.

Sutton, R. S. & Barto, A. G. (1998). Reinforcement Learning: An Introduction. (Cambridge, MA: MIT Press).

Sutton, R. S., Mcallester, D., Singh, S., & Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. Adv. Neural Inf. Process. Syst. pp. 1057–1063. http://papers.nips.cc/paper/1713-policy-gradient-methods-for-reinforcement-learning-with-function-approximation.pdf.

Takahashi, Y. K., Roesch, M. R., Wilson, R. C., Toreson, K., O'Donnell, P., Niv, Y., & Schoenbaum, G. (2011). Expectancy-related changes in firing of dopamine neurons depend on orbitofrontal cortex. Nat Neurosci 14, 1590–1597. http://dx.doi.org/10.1038/nn.2957.

The Theano Development Team (2016). Theano: A Python framework for fast computation of mathematical expressions. arXiv:1605.02688. http://arxiv.org/abs/1605.02688.

Todd, M. T., Niv, Y., & Cohen, J. D. (2008). Learning to use working memory in partially observable environments through dopaminergic reinforcement. Adv. Neural Inf. Process. Syst. http://papers.nips.cc/paper/3508-learning-to-use-working-memory-in-partially-observable-environments-through-dopaminergic-reinforcement.pdf.

Turner, R. S. & Desmurget, M. (2010). Basal ganglia contributions to motor control: A vigorous tutor. Curr. Opin. Neurobiol. 20, 704–716. http://dx.doi.org/10.1016/j.conb.2010.08.022.

Urbanczik, R. & Senn, W. (2009). Reinforcement learning in populations of spiking neurons. Nat Neurosci 12, 250–252. http://dx.doi.org/10.1038/nn.2264.

Wallis, J. D. (2007). Orbitofrontal cortex and its contribution to decision-making. Annu. Rev. Neurosci. 30, 31–56. http://dx.doi.org/10.1146/annurev.neuro.30.051606.094334.

Wang, X.-J. (2002). Probabilistic decision making by slow reverrberation in cortical circuits. Neuron 36, 955–968. http://dx.doi.org/10.1016/S0896-6273(02)01092-9.

Wang, X.-J. (2008). Decision making in recurrent neuronal circuits. Neuron 60, 215–234. http://dx.doi.org/10.1016/j.neuron.2008.09.034.

Wei, Z. & Wang, X.-J. (2015). Confidence estimation as a stochastic process in a neurodynamical system of decision making. J. Neurophysiol. 114, 99–113. http://dx.doi.org/10.1152/jn.00793.2014.

Wierstra, D., Förster, A., Peters, J., & Schmidhuber, J. (2009). Recurrent policy gradients. Log. Jnl IGPL 18, 620–634. http://dx.doi.org/10.1093/jigpal/jzp049.

Williams, R. J. (1992). Simple statistical gradient-following methods for connectionist reinforcement learning. Mach. Learn. 8, 229–256. http://dx.doi.org/10.1007/BF00992696.

Wong, K.-F. & Wang, X.-J. (2006). A recurrent network mechanism of time integration in perceptual decisions. J. Neurosci. 26, 1314–1328. http://dx.doi.org/10.1523/JNEUROSCI.3733-05.2006.

Xu, K., Ba, J. L., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R. S., & Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. Proc. 32 nd Int. Conf. Mach. Learn. http://jmlr.org/proceedings/papers/v37/xuc15.pdf.

Yamins, D. L. K., Hong, H., Cadieu, C. F., Solomon, E. A., Seibert, D., & DiCarlo, J. J. (2014). Performance-optimized hierarchical models predict neural responses in higher visual cortex. Proc. Natl. Acad. Sci. U. S. A. 111, 8619–24. http://dx.doi.org/10.1073/pnas.1403112111.

Zaremba, W. & Sutskever, I. (2016). Reinforcement Learning Neural Turing Machines. arXiv:1505.00521. http://arxiv.org/abs/1505.00521.

Zipser, D. & Andersen, R. A. (1988). A back-propagation programmed network that simulates response properties of a subset of posterior parietal neurons. Nature 331, 679–684. http://dx.doi.org/10.1038/331679a0.

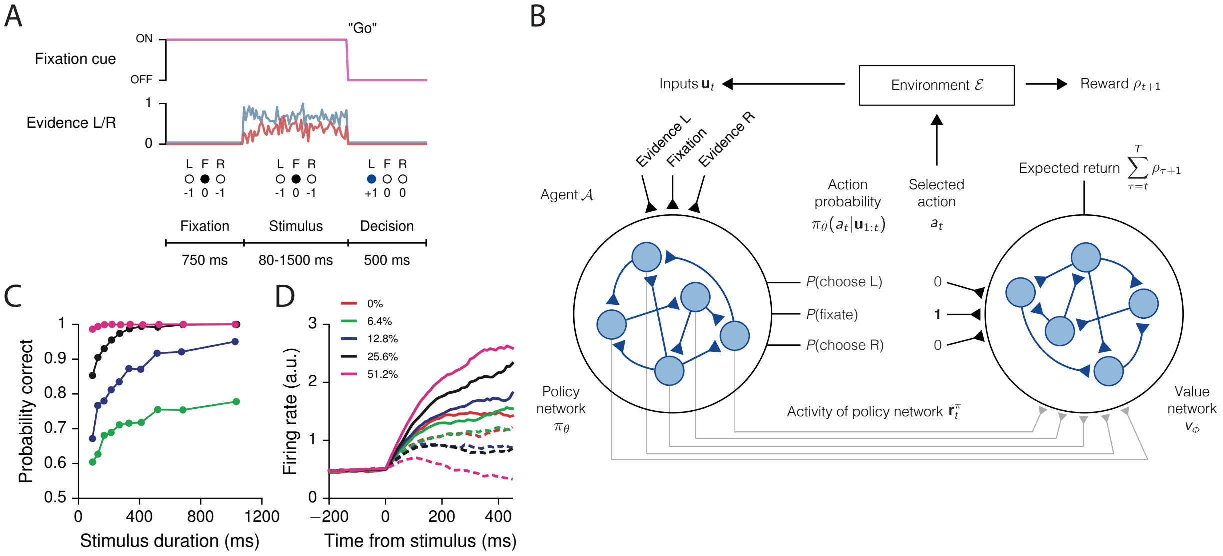
A
Fixation cue
ON
OFF
"Go"
Evidence L/R
1
0
L F R
○ ● ○
-1 0 -1
L F R
○ ● ○
-1 0 -1
L F R
● ○ ○
+1 0 0
Fixation
Stimulus
Decision
750 ms
80-1500 ms
500 ms
B
Inputs $\mathbf{u}_t$
Environment $\mathcal{E}$
Reward ρ_{t+1}
Evidence L
Fixation
Evidence R
Agent $\mathcal{A}$
Action probability $\pi_\theta(a_t|\mathbf{u}_{1:t})$
Selected action a_t
Expected return $\sum_{\tau=t}^{T}\rho_{\tau+1}$
P(choose L)
0
P(fixate)
1
P(choose R)
0
Policy network π_θ
Activity of policy network $\mathbf{r}_t^\pi$
Value network v_ϕ
C
Probability correct
1
0.9
0.8
0.7
0.6
0.5
0
400
800
1200
Stimulus duration (ms)
D
0%
6.4%
12.8%
25.6%
51.2%
Firing rate (a.u.)
3
2
1
0
−200
0
200
400
Time from stimulus (ms)

Figure 1

Figure 1. Recurrent neural networks for reinforcement learning. (**A**) Task structure for a simple perceptual decision-making task with variable stimulus duration. The agent must maintain fixation ($a_t = \mathrm{F}$) until the go cue, which indicates the start of a decision period during which choosing the correct response ($a_t = \mathrm{L}$ or $a_t = \mathrm{R}$) results in a positive reward. The agent receives zero reward for responding incorrectly, while breaking fixation early results in an aborted trial and negative reward. (**B**) At each time t the agent selects action a_t according to the output of the policy network π_θ, which can depend on all past and current inputs $\mathbf{u}_{1:t}$ provided by the environment. In response, the environment transitions to a new state and provides reward ρ_{t+1} to the agent. The value network v_ϕ uses the selected action and the activity of the policy network $\mathbf{r}_t^\pi$ to predict future rewards. All the weights shown are plastic, i.e., trained by gradient descent. (**C**) Performance of the network trained for the task in (A), showing the percent correct by stimulus duration, for different coherences (the difference in strength of evidence for L and R). (**D**) Neural activity of an example policy network unit, sorted by coherence and aligned to the time of stimulus onset. Solid lines are for positive coherence, dashed for negative coherence.

Figure supplement 1. Learning curves for the simple perceptual decision-making task.

Figure supplement 2. Output of the value network indicating expected reward.

Figure supplement 3. Reaction-time version of the perceptual decision-making task, in which the go cue coincides with the onset of stimulus, allowing the agent to choose when to respond.

Figure supplement 4. Learning curves for the reaction-time version of the simple perceptual decision-making task.

Figure supplement 5. Learning curves for the simple perceptual decision-making task with a linear readout of the policy network as the baseline.

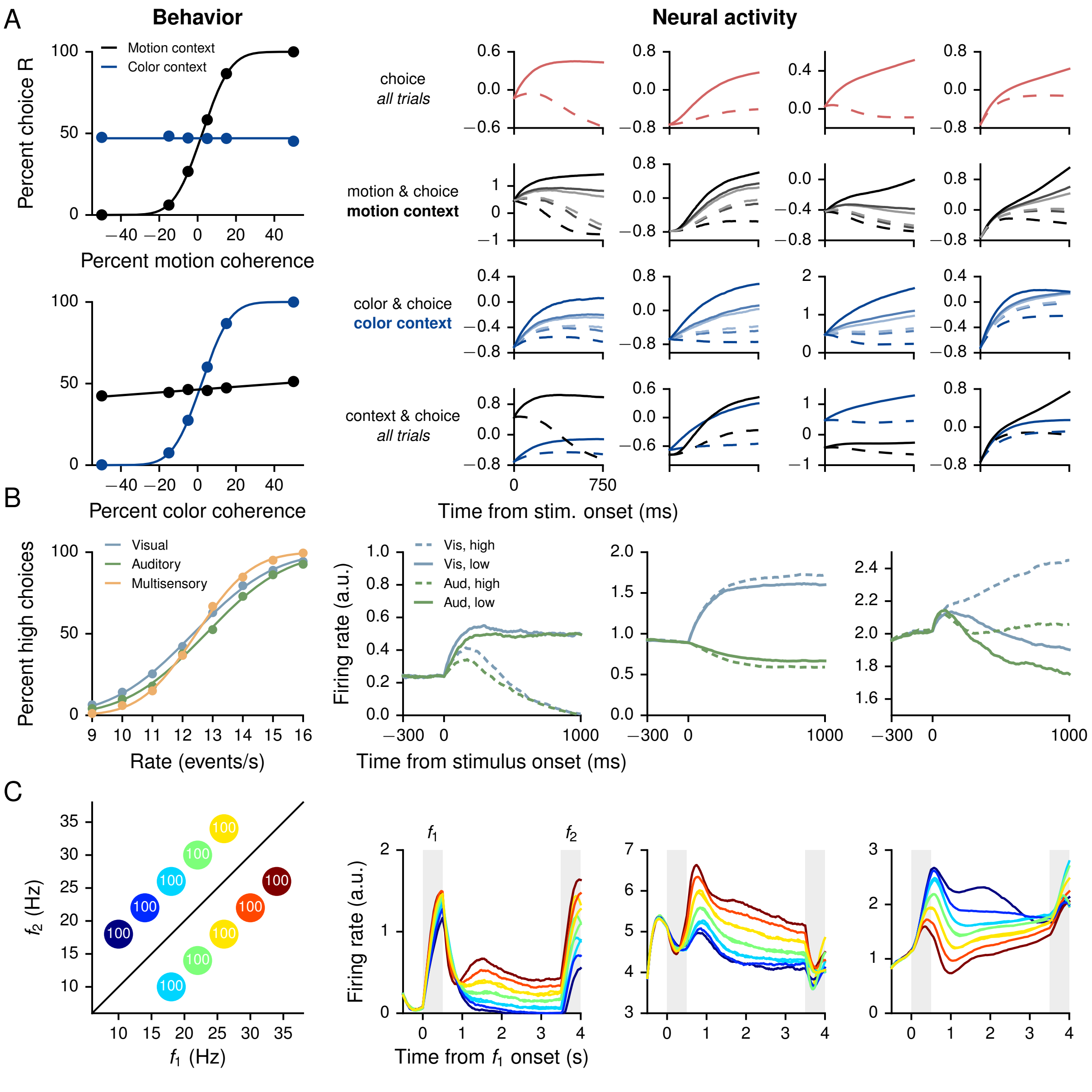
A
Behavior
Neural activity
Motion context
Color context
Percent choice R
Percent motion coherence
Percent color coherence
choice
all trials
motion & choice
motion context
color & choice
color context
context & choice
all trials
Time from stim. onset (ms)
0
750
B
Visual
Auditory
Multisensory
Percent high choices
Rate (events/s)
Vis, high
Vis, low
Aud, high
Aud, low
Firing rate (a.u.)
Time from stimulus onset (ms)
C
f_2 (Hz)
f_1 (Hz)
100
f_1
f_2
Firing rate (a.u.)
Time from f_1 onset (s)

Figure 2

Figure 2. Performance and neural activity of RNNs trained for "simple" cognitive tasks in which the correct response depends only on the task condition. Left column shows behavioral performance, right column shows mixed selectivity for task parameters of example units in the policy network. (**A**) Context-dependent integration task (Mante et al., 2013). Left: Psychometric curves show the percentage of R choices as a function of the signed "motion" and "color" coherences. Right: Normalized firing rates of examples units sorted by different combinations of task parameters exhibit mixed selectivity. Firing rates were normalized by mean and standard deviation computed over the responses of all units, times, and trials. (**B**) Multisensory integration task (Raposo et al., 2012, 2014). Left: Psychometric curves show the percentage of high choices as a function of the event rate, for visual only (blue), auditory only (green), and multisensory (orange) trials. Improved performance on multisensory trials shows that the network learns to combine the two sources of information in accordance with Equation 13. Right: Sorted activity on visual only and auditory only trials for units selective for choice (high vs. low, left), modality (visual vs. auditory, middle), and both (right). (**C**) Parametric working memory task (Romo et al., 1999). Left: Percentage of correct responses for different combinations of f_1 and f_2. The conditions are colored here and in the right panels according to the first stimulus (base frequency) f_1; due to the overlap in the values of f_1, the 10 task conditions are represented by 7 distinct colors. Right: Activity of example policy network units sorted by f_1. The first two units are positively tuned to f_1 during the delay period, while the third unit is negatively tuned.

Figure supplement 1. Learning curves for the context-dependent integration task (A).

Figure supplement 2. Learning curves for the multisensory integration task (B).

Figure supplement 3. Learning curves for the parametric working memory task (C).

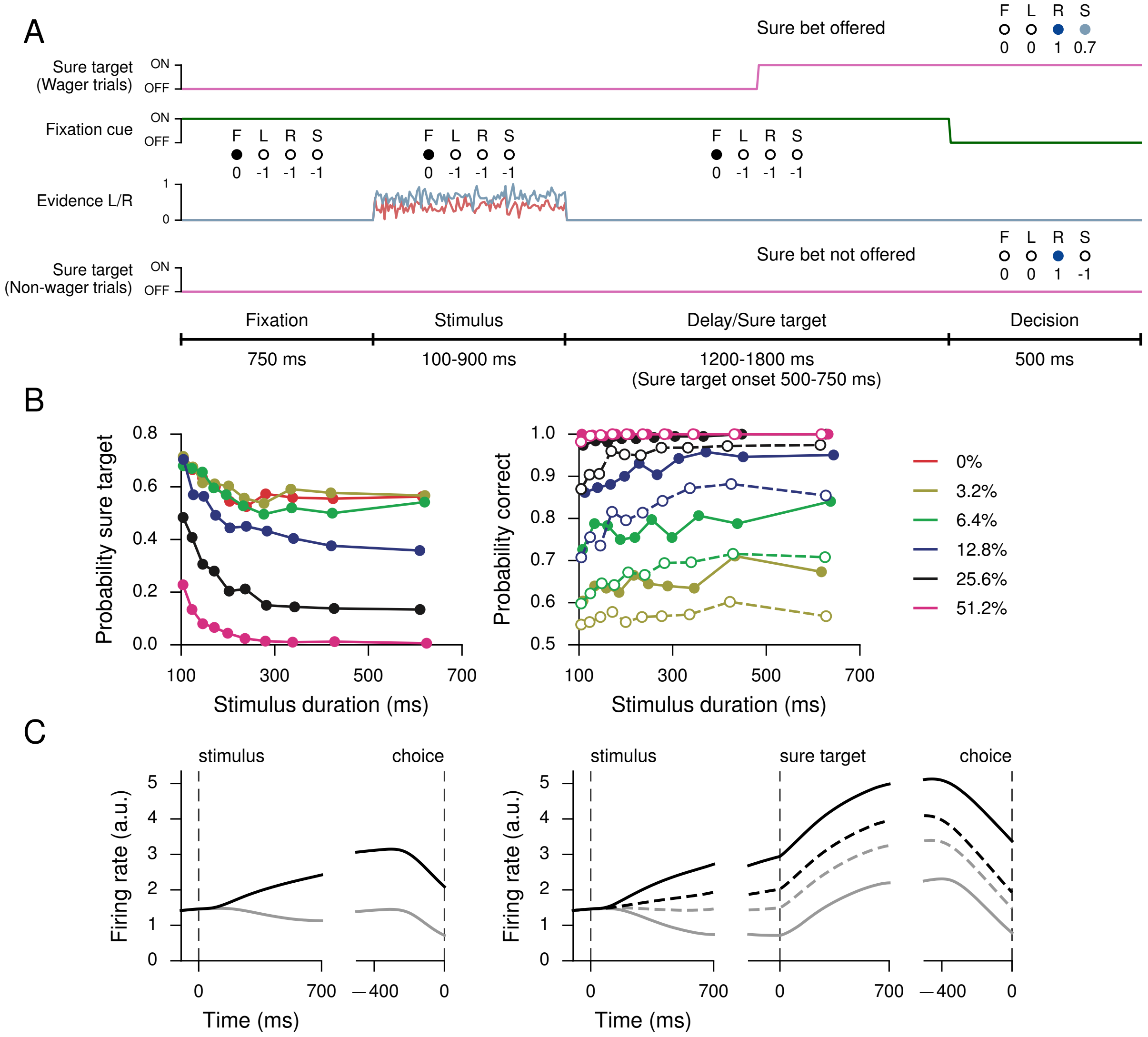
A
Sure bet offered
F L R S
0 0 1 0.7
Sure target (Wager trials)
ON
OFF
Fixation cue
ON
OFF
F L R S
0 -1 -1 -1
F L R S
0 -1 -1 -1
F L R S
0 -1 -1 -1
Evidence L/R
1
0
Sure bet not offered
F L R S
0 0 1 -1
Sure target (Non-wager trials)
ON
OFF
Fixation
Stimulus
Delay/Sure target
Decision
750 ms
100-900 ms
1200-1800 ms (Sure target onset 500-750 ms)
500 ms
B
Probability sure target
0.0
0.2
0.4
0.6
0.8
100
300
500
700
Stimulus duration (ms)
Probability correct
0.5
0.6
0.7
0.8
0.9
1.0
100
300
500
700
Stimulus duration (ms)
0%
3.2%
6.4%
12.8%
25.6%
51.2%
C
stimulus
choice
Firing rate (a.u.)
0
1
2
3
4
5
0
700
−400
0
Time (ms)
stimulus
sure target
choice
Firing rate (a.u.)
0
1
2
3
4
5
0
700
0
700
−400
0
Time (ms)

Figure 3

Figure 3. Perceptual decision-making task with postdecision wagering, based on Kiani & Shadlen (2009). (**A**) Task structure. On a random half of the trials, a sure option is presented during the delay period, and on these trials the network has the option of receiving a smaller (compared to correctly choosing L or R) but certain reward by choosing the sure option (S). The stimulus duration, delay, and sure target onset time are the same as in Kiani & Shadlen (2009). (**B**) Probability of choosing the sure option (left) and probability correct (right) as a function of stimulus duration, for different coherences. Performance is higher for trials on which the sure option was offered but waived in favor of L or R (filled circles, solid), compared to trials on which the sure option was not offered (open circles, dashed). (**C**) Activity of an example policy network unit for non-wager (left) and wager (right) trials, sorted by whether the presented evidence was toward the unit's preferred (black) or nonpreferred (gray) target as determined by activity during the stimulus period on all trials. Dashed lines show activity for trials in which a sure option was chosen.

Figure supplement 1. Learning curves.

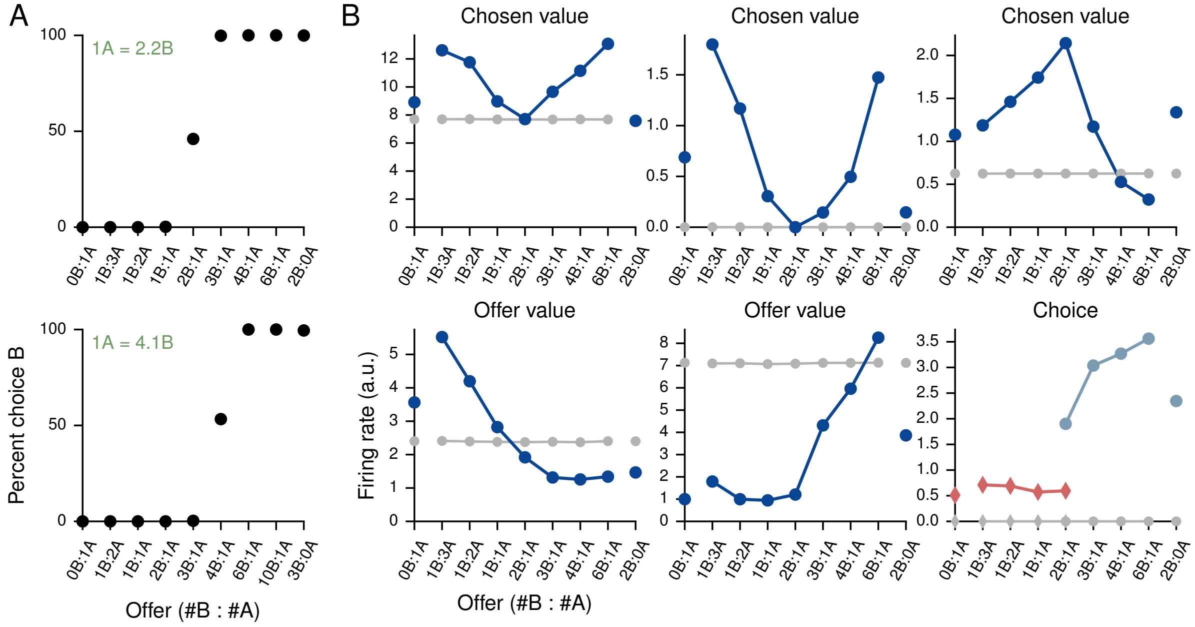

Figure 4. Value-based economic choice task (Padoa-Schioppa & Assad, 2006). (**A**) Choice pattern when the reward contingencies are indifferent for roughly 1 "juice" of A and 2 "juices" of B (upper) or 1 juice of A and 4 juices of B (lower). (**B**) Mean activity of example value network units during the pre-choice period, defined here as the period 500 ms before the decision, for the 1A = 2B case. Units in the value network exhibit diverse selectivity as observed in the monkey orbitofrontal cortex. For "choice" (last panel), trials were separated into choice A (red diamonds) and choice B (blue circles).

Figure supplement 1. Fit of cumulative Gaussian to the choice pattern in (A), upper.

Figure supplement 2. Fit of cumulative Gaussian to the choice pattern in (A), lower.

Figure supplement 3. Learning curves.

Parameter	**Symbol**	**Default value**
Learning rate	η	0.004
Maximum gradient norm	Γ	1
Size of policy/value network	N	100
Connection probability (policy network)	p_c^π	0.1
Connection probability (value network)	p_c^v	1
Time step	Δt	10 ms
Unit time constant	τ	100 ms
Recurrent noise	σ_{rec}^2	0.01
Initial spectral radius for recurrent weights	ρ_0	2
Number of trials per gradient update	N_{trials}	# of task conditions

Table 1. Parameters for reward-based recurrent neural network training. Unless noted otherwise in the text, networks were trained and run with the parameters listed here.

σ_{visual}	σ_{auditory}	$\sigma_{\text{multisensory}}$	$\dfrac{1}{\sigma^2_{\text{visual}}} + \dfrac{1}{\sigma^2_{\text{auditory}}}$	$\dfrac{1}{\sigma^2_{\text{multisensory}}}$
2.124	**2.099**	**1.451**	**0.449**	**0.475**
2.107	2.086	1.448	0.455	0.477
2.276	2.128	1.552	0.414	0.415
2.118	2.155	1.508	0.438	0.440
2.077	2.171	1.582	0.444	0.400
2.088	2.149	1.480	0.446	0.457

Table 2. Psychophysical thresholds σ_{visual}, σ_{auditory}, and $\sigma_{\text{multisensory}}$ obtained from fits of cumulative Gaussian functions to the psychometric curves in visual only, auditory only, and multisensory trials in the multisensory integration task, for 6 networks trained from different random initializations (first row, bold: network from main text, cf. Figure 2B). The last two columns show evidence of "optimal" multisensory integration according to Equation 13 (Raposo et al., 2012).

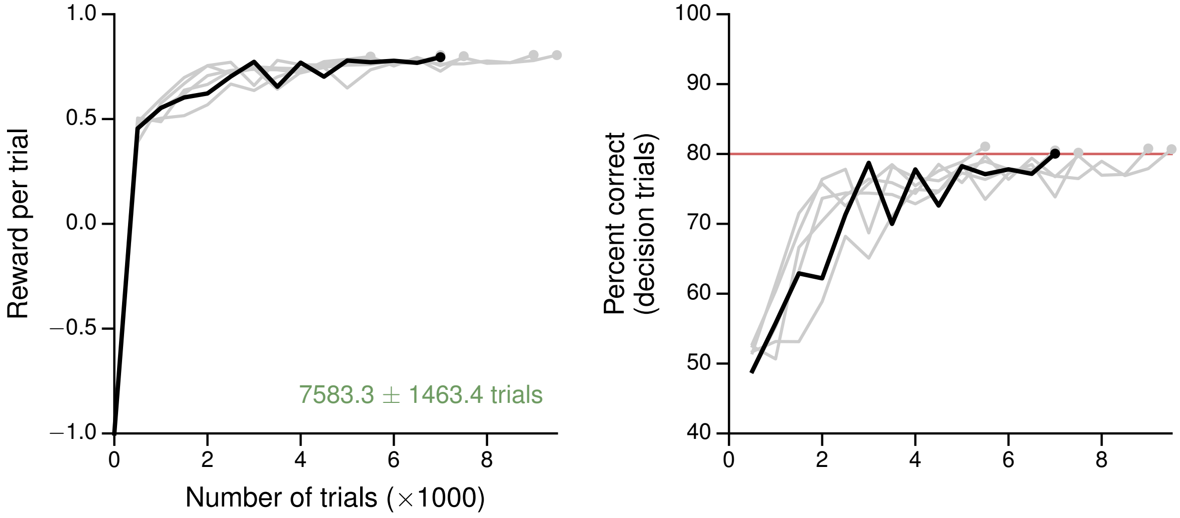

Figure 1—figure supplement 1. Learning curves for the simple perceptual decision-making task. (**A**) Average reward per trial. Black indicates the network realization shown in the main text, gray additional realizations, i.e., trained with different random number generator seeds. (**B**) Percent correct, for trials on which the network made a decision (≥ 99% required for termination). Red: target performance (when training was terminated).

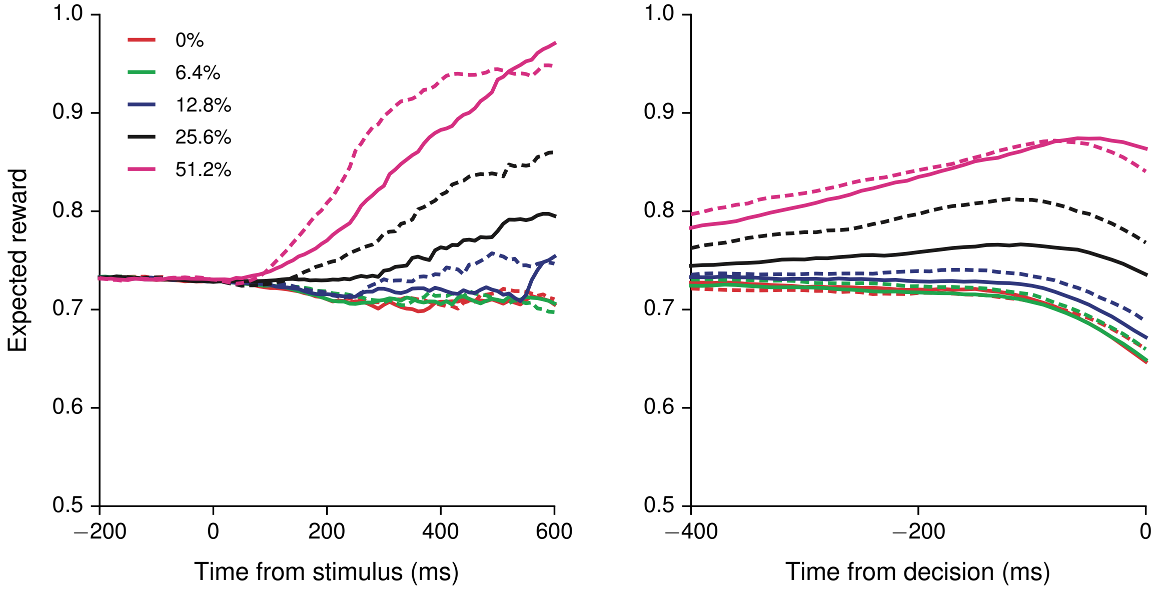

Figure 1—figure supplement 2. Output of the value network (expected reward) aligned to (**A**) stimulus onset and (**B**) choice, sorted by coherence for positive (solid) and negative (dashed) coherences.

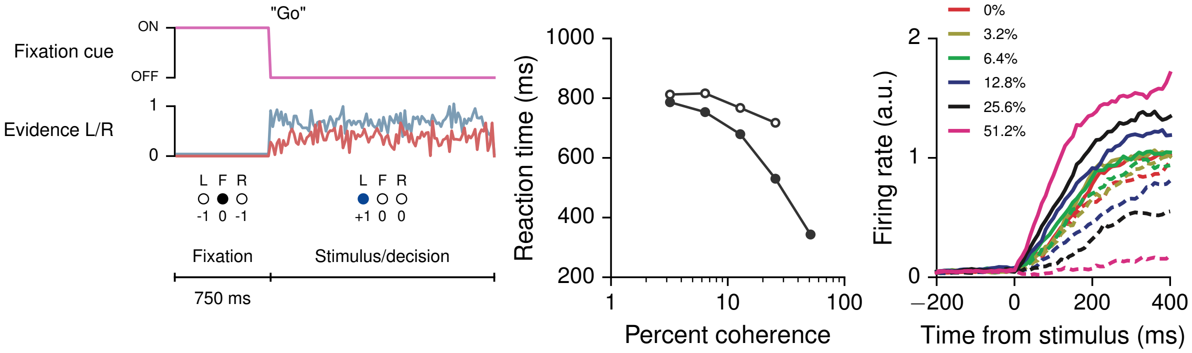

Figure 1—figure supplement 3. (**A**) Task structure for the reaction-time version of the simple perceptual decision-making task, in which the agent can choose to respond any time after the onset of stimulus. (**B**) Reaction time as a function of coherence for correct (solid circles) and error (open circles) trials. (**C**) Neural activity of an example policy network unit, sorted by the coherence (the difference in strength of evidence for L and R) and aligned to the time of stimulus onset. Each trial ends when the network breaks fixation.

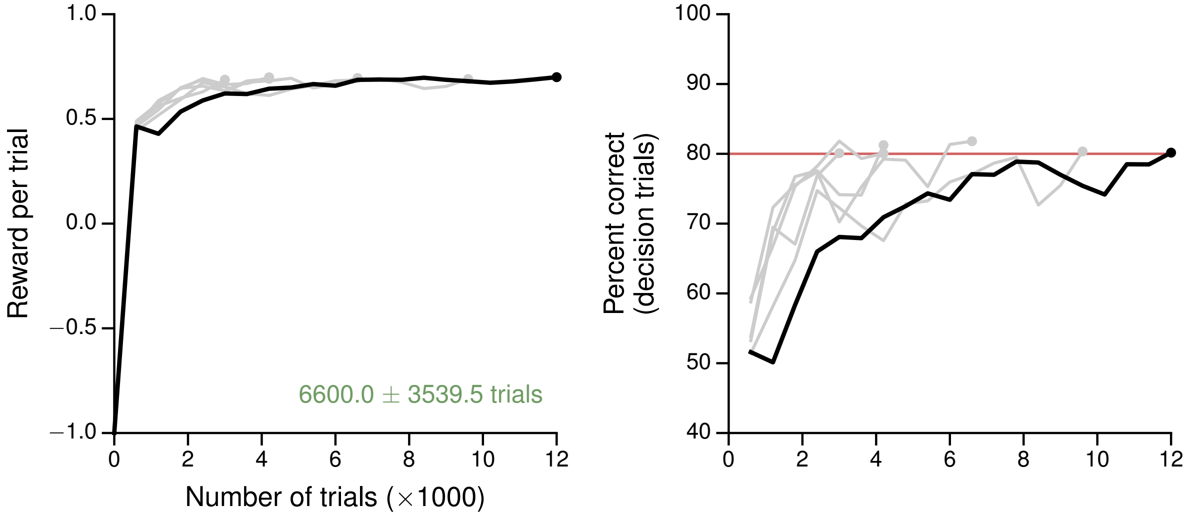

Figure 1—figure supplement 4. Learning curves for the reaction-time version of the simple perceptual decision-making task. (**A**) Average reward per trial. Black indicates the network realization shown in the main text, gray additional realizations, i.e., trained with different random number generator seeds. (**B**) Percent correct, for trials on which the network made a decision (≥ 99% required for termination). Red: target performance (when training was terminated).

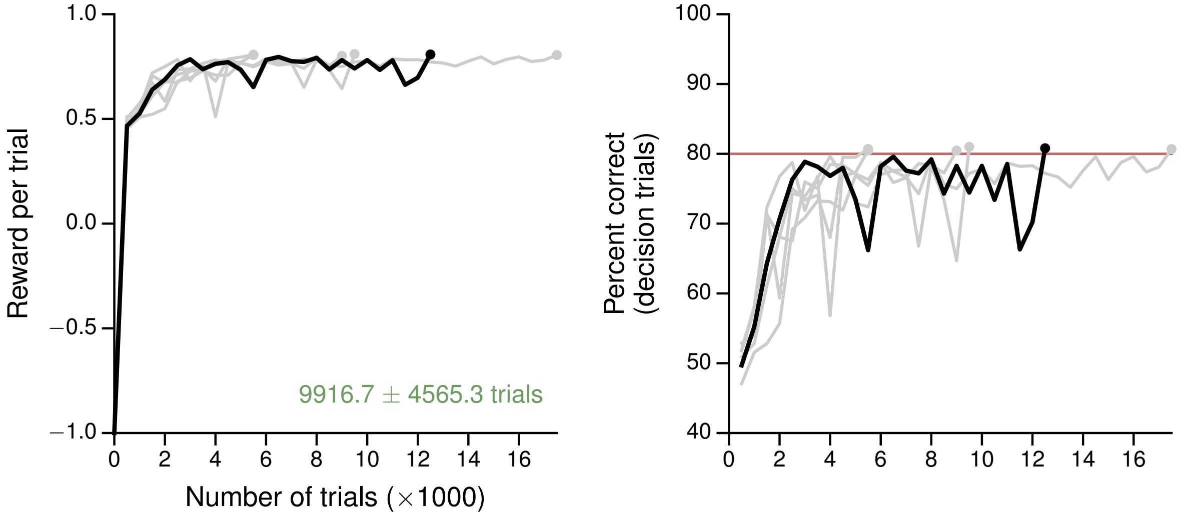

Figure 1—figure supplement 5. Learning curves for the simple perceptual decision-making task with a linear readout of the policy network as the baseline. (**A**) Average reward per trial. Black indicates the network realization shown in the main text, gray additional realizations, i.e., trained with different random number generator seeds. (**B**) Percent correct, for trials on which the network made a decision (≥ 99% required for termination). Red: target performance (when training was terminated).

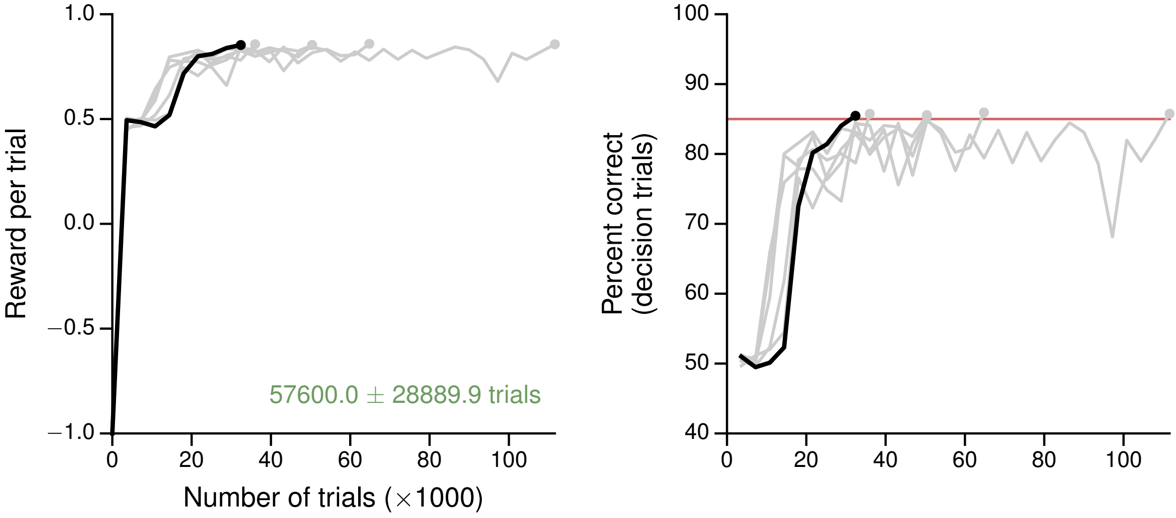

Figure 2—figure supplement 1. Learning curves for the context-dependent integration task. (**A**) Average reward per trial. Black is for the network realization in the main text, gray for additional realizations, i.e., trained with different random number generator seeds. (**B**) Percent correct, for trials on which the network made a decision (≥ 99% required for termination). Red: target performance (when training was terminated).

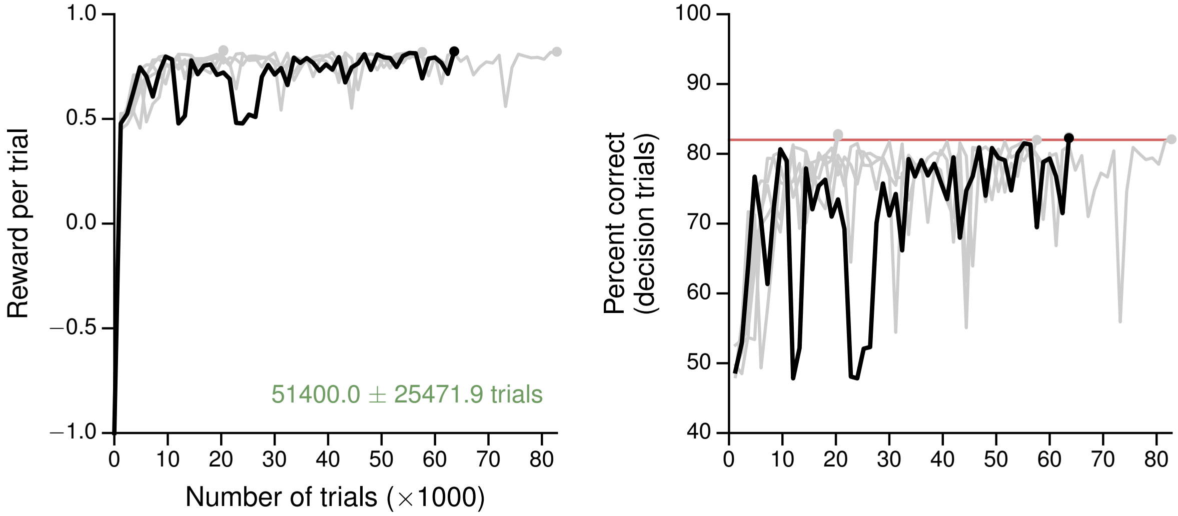

Figure 2—figure supplement 2. Learning curves for the multisensory integration task. (**A**) Average reward per trial. Black indicates the network realization shown in the main text, gray additional realizations, i.e., trained with different random number generator seeds. (**B**) Percent correct, for trials on which the network made a decision (≥ 99% required for termination). Red: target performance (when training was terminated).

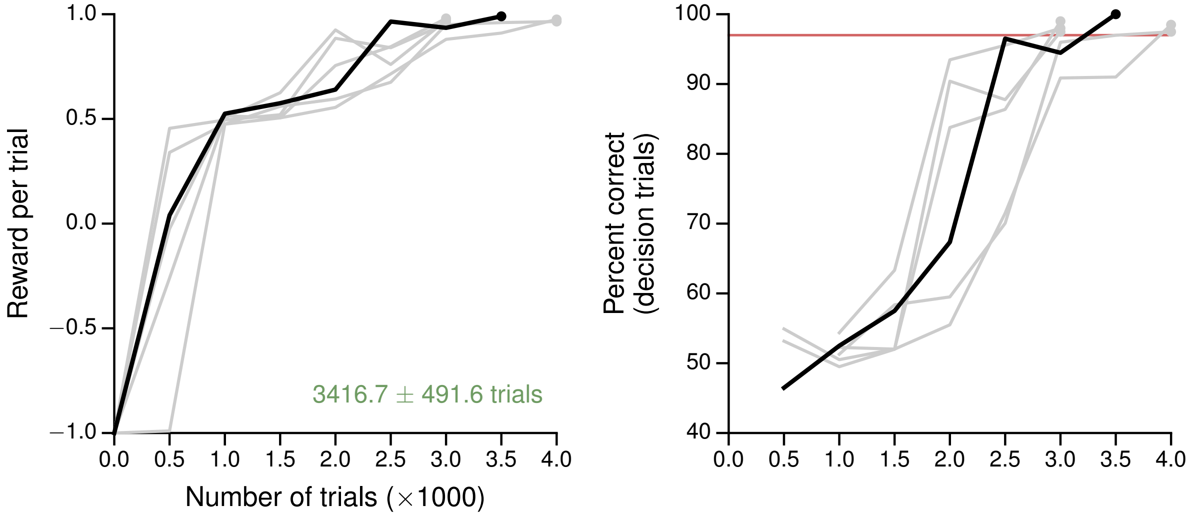

Figure 2—figure supplement 3. Learning curves for the parametric working memory task. (**A**) Average reward per trial. Black indicates the network realization shown in the main text, gray additional realizations, i.e., trained with different random number generator seeds. (**B**) Percent correct, for trials on which the network made a decision (≥ 99% required for termination). Red: target performance (when training was terminated).

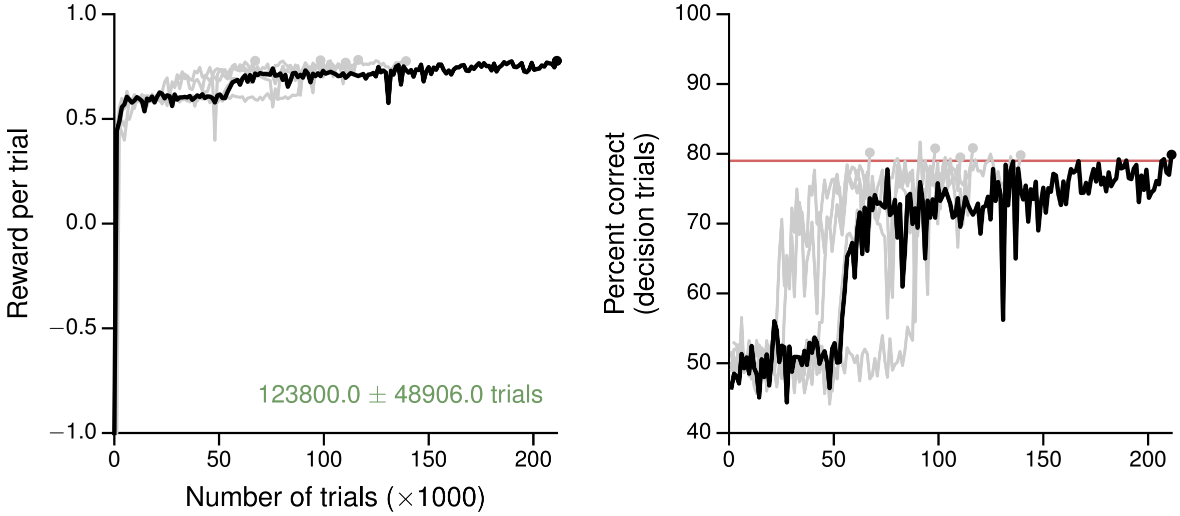

Figure 3—figure supplement 1. Learning curves for the postdecision wager task. (**A**) Average reward per trial. Black indicates the network realization shown in the main text, gray additional realizations, i.e., trained with different random number generator seeds. (**B**) Percent correct, for trials on which the network made a decision (≥ 99% required for termination). Red: target performance when the sure bet was accepted between 40-50% of the time.

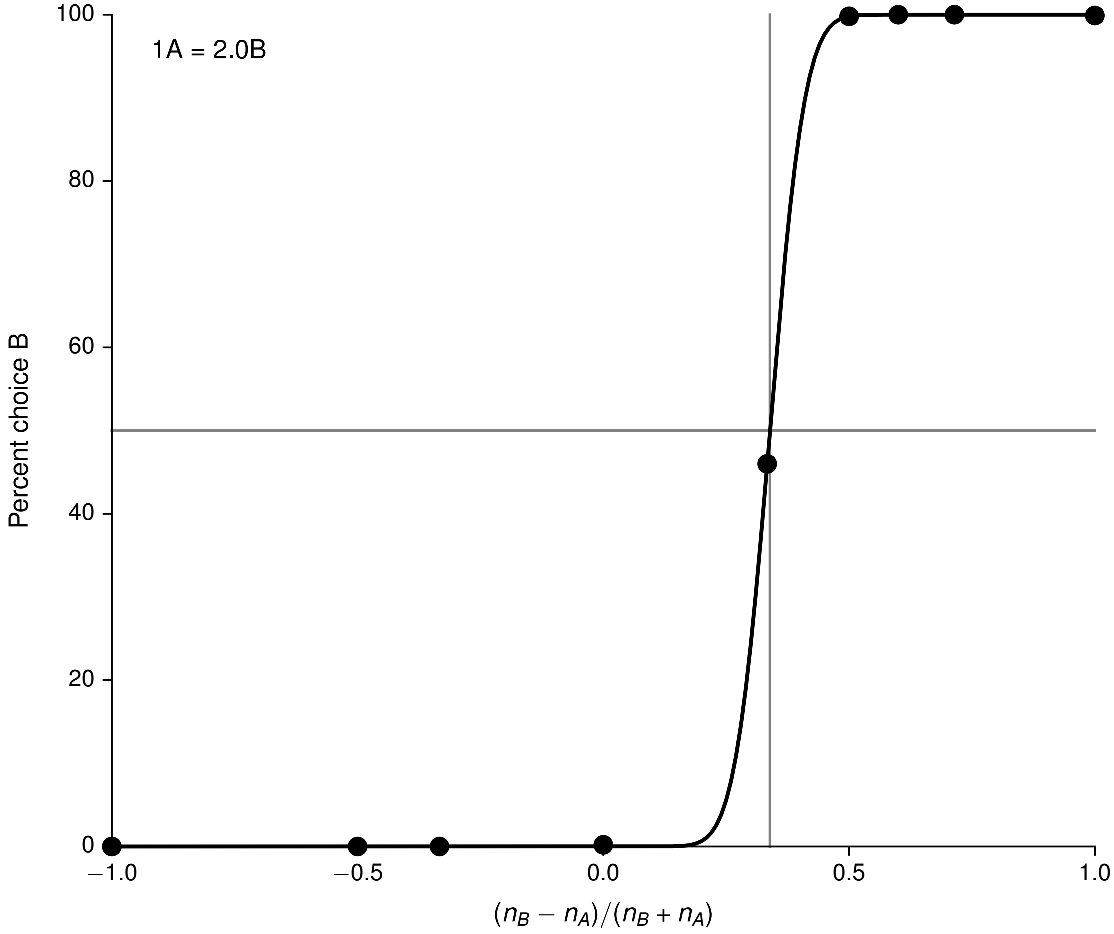

Figure 4—figure supplement 1. Fit of cumulative Gaussian with parameters μ, σ to the choice pattern in Figure 4 (upper), and the deduced indifference point $n^*_B/n^*_A = (1+\mu)/(1-\mu)$.

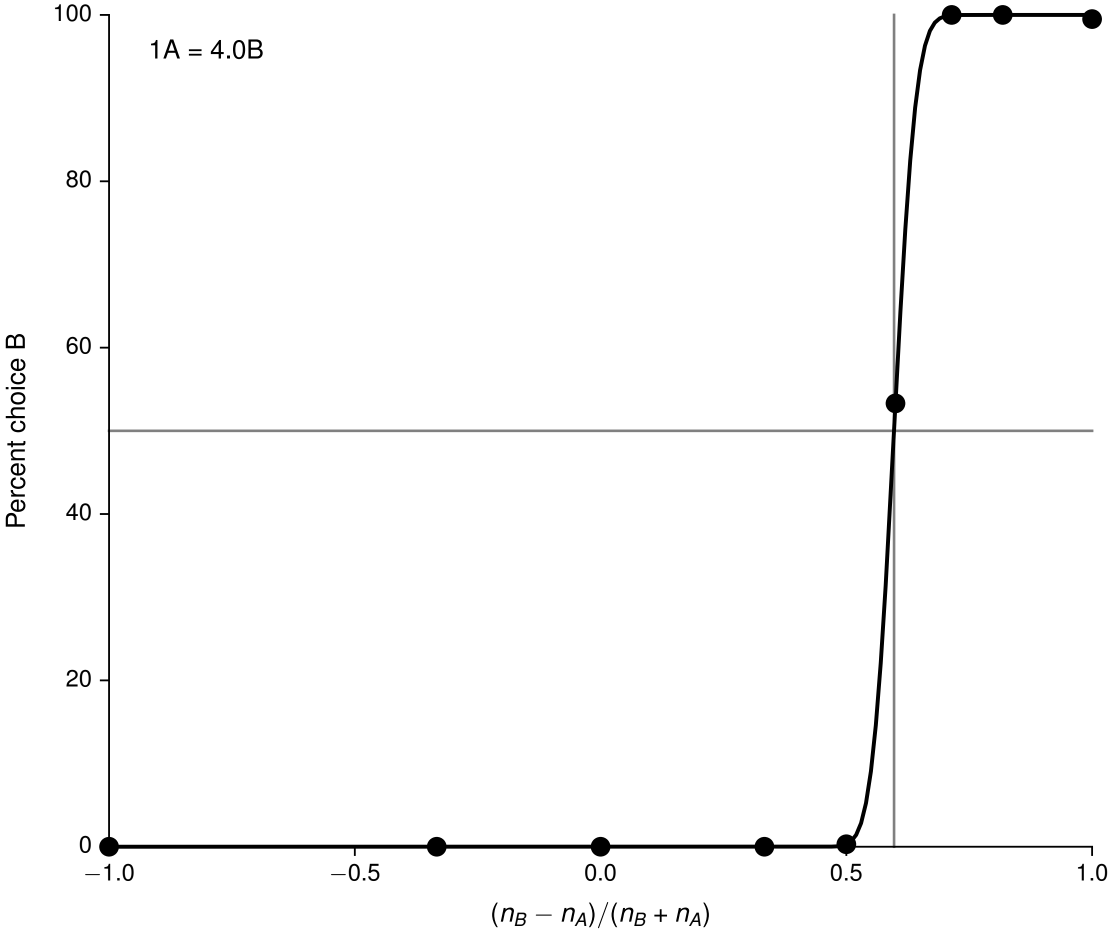

Figure 4—figure supplement 2. Fit of cumulative Gaussian with parameters μ, σ to the choice pattern in Figure 4A (lower), and the deduced indifference point $n_B^*/n_A^* = (1+\mu)/(1-\mu)$.

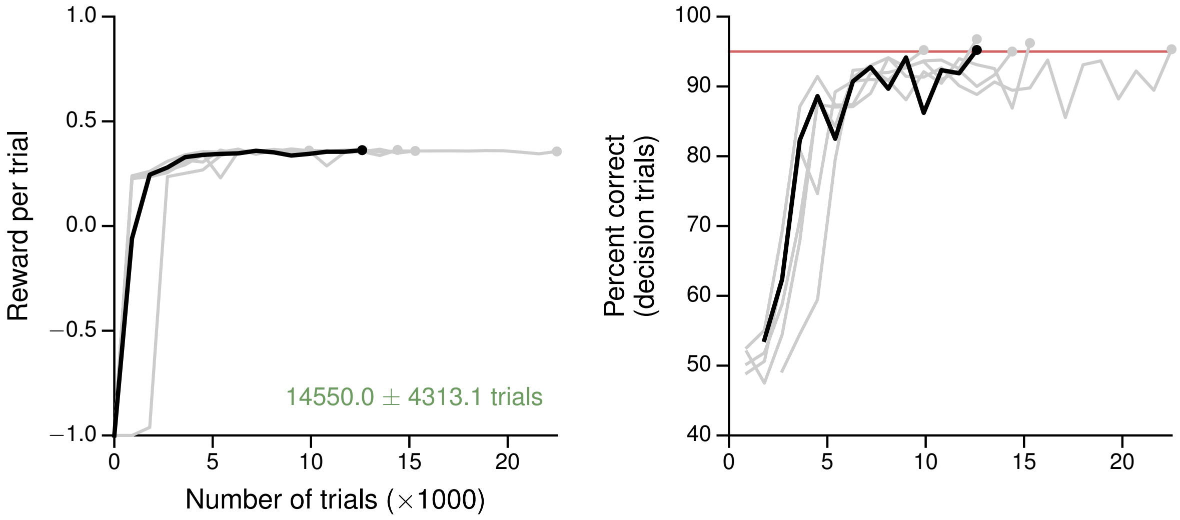

Figure 4—figure supplement 3. Learning curves for the value-based economic choice task. (**A**) Average reward per trial. Black indicates the network realization shown in the main text, gray additional realizations, i.e., trained with different random number generator seeds. (**B**) Percentage of trials on which the network chose the option that resulted in greater (or equal) reward, for trials where the network made a decision (≥ 99% required for termination). Note this is conceptually different from the previous tasks, where "correct" depends on the sensory inputs, not the rewards. Red: target performance (when training was terminated).