COGNITIVE NEUROSCIENCE

# Computation noise promotes zero-shot adaptation to uncertainty during decision-making in artificial neural networks

**Charles Findling[1,2]* and Valentin Wyart[1,3,4]***



**Random noise in information processing systems is widely seen as detrimental to function. But despite the large trial-to-trial variability of neural activity, humans show a remarkable adaptability to conditions with uncertainty during goal-directed behavior. The origin of this cognitive ability, constitutive of general intelligence, remains elusive. Here, we show that moderate levels of computation noise in artificial neural networks promote zero-shot generalization for decision-making under uncertainty. Unlike networks featuring noise-free computations, but like human participants tested on similar decision problems (ranging from probabilistic reasoning to reversal learning), noisy networks exhibit behavioral hallmarks of optimal inference in uncertain conditions entirely unseen during training. Computation noise enables this cognitive ability jointly through "structural" regularization of network weights during training and "functional" regularization by shaping the stochastic dynamics of network activity after training. Together, these findings indicate that human cognition may ride on neural variability to support adaptive decisions under uncertainty without extensive experience or engineered sophistication.**

## INTRODUCTION

Extracting signal from noise is seen as a core feature of efficient information processing systems, from gravitational-wave detectors to neural networks. In this context, noise is usually defined as irrelevant input that should be filtered out to improve signal detection. But beyond this input noise, brains process and respond to input with a large internal variability (*1*). This computation noise has wide-ranging impacts on human cognition (*2*, *3*), from fluctuations in the perception of weak or ambiguous sensory stimuli (*4*–*6*) to exploratory decisions during reward-guided behavior (*7*, *8*). Existing neuroscience research considers this internal variability as a hard constraint on neural information processing systems, in that the brain has evolved to cope with using efficient coding strategies (*9*–*12*).

A separate line of research in psychology has demonstrated the remarkable adaptability of human cognition to a wide range of conditions involving uncertainty (which we will refer to as "uncertain conditions") without extensive training in each of them. Competing theories postulate that humans have developed general-purpose strategies, either heuristics (*13*, *14*) or normative computations (*15*, *16*), to respond efficiently to uncertainty. In both cases, these strategies require prior experience with uncertain conditions, and they are thought to emerge despite the large internal variability of neural activity.

However, internal variability does not only constitute a nuisance for information processing systems. It is well-known that introducing variability during the training of artificial neural networks, e.g., by randomly inactivating some of their units, reduces their natural tendency to overfitting (*17*) and that adding stochasticity to nonlinear dynamical systems changes their properties in several nontrivial ways, e.g., by allowing transitions between otherwise stable states (*18*, *19*). These two effects can be seen as distinct forms of regularization: (i) structural regularization by tuning the connection weights of artificial neural networks and (ii) functional regularization by shaping the dynamics of stochastic nonlinear systems. These two forms of regularization are observed across different systems shaped by different sources of variability, variability (e.g., the random inactivation of units) that often does not resemble the computation noise observed in humans (*2*, *3*).

Here, we hypothesized that computation noise may promote the high adaptability of human cognition to uncertainty by providing both structural and functional regularization in neural circuits. To test this hypothesis, we used recurrent neural networks (RNNs) as flexible models that we could train to perform cognitive tasks involving different sources of uncertainty where humans feature substantial computation noise. Using artificial cognitive models enabled us to causally investigate the potential functions of computation noise by comparing RNNs featuring either exact or noisy computations (Fig. 1) in two widely used experimental frameworks for studying adaptive learning and decision-making in humans (*20*, *21*). We trained the networks on a task A using reinforcement learning (RL) and then tested them on a more challenging variant A* of the same task which requires taking into account a source of uncertainty absent from task A. Across decision problems, we found that computation noise, unlike other regularization mechanisms, confers near-optimal adaptability to different types of uncertainty and generates behavioral variability whose signatures are notably similar to those described in humans.

## RESULTS

### Zero-shot performance in the weather prediction task

The first task A that we considered requires learning probabilistic associations between stimuli and rewarded actions (Fig. 2A). We trained the weights of exact and noisy RNNs on this task. Computation noise was modeled as additive normally distributed noise affecting the activity of each recurrent unit in the network (see Fig. 1

[1]Laboratoire de Neurosciences Cognitives et Computationnelles, Institut National de la Santé et de la Recherche Médicale (Inserm), Paris, France. [2]Département des Neurosciences Fondamentales, Université de Genève, Geneva, Switzerland. [3]Département d'Études Cognitives, École Normale Supérieure, Université PSL, Paris, France. [4]Institut du Psychotraumatisme de l'Enfant et de l'Adolescent, Conseil Départemental Yvelines et Hauts-de-Seine, Versailles, France.
*Corresponding author. Email: charles.findling@internationalbrainlab.org (C.F.); valentin.wyart@inserm.fr (V.W.)

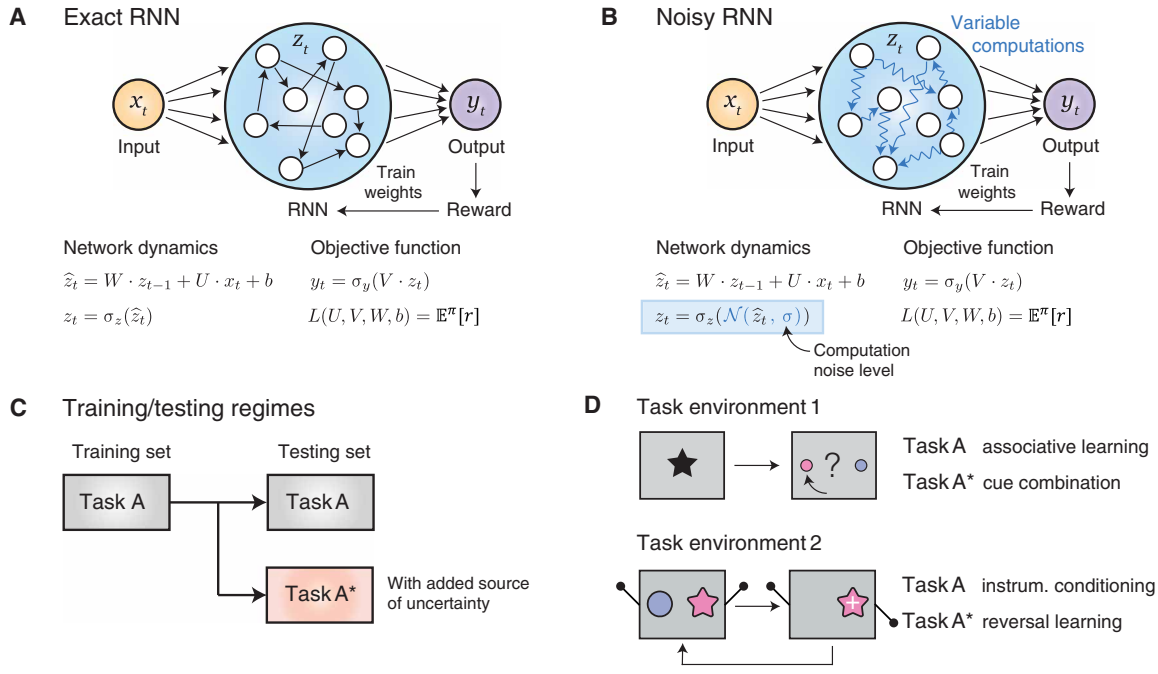


**Fig. 1. RNNs and training/testing regimes.** The decision-making RNN is fed with input $x_t$, which is combined with the previous recurrent activity $z_{t-1}$ and passed through a nonlinear activation function $\sigma_z(\cdot)$ to obtain the updated recurrent activity $z_t$. The output (decision policy) $y_t$ of the RNN is obtained from $z_t$ and passed through a softmax function $\sigma_y(\cdot)$ to choose an action. (**A**) RNN with exact (noise-free) computations. (**B**) RNN with noisy computations. The recurrent network updates are now corrupted by zero-mean normally distributed noise of SD $\sigma$. (**C**) Training and testing regimes. The weights of the decision-making RNN are trained using backpropagation on a first task A. The trained weights are then frozen, and the RNN is tested either on task A or on a variant of the task A (task A*) with an added source of uncertainty. (**D**) Studied task environments. Task environment 1: The RNN is presented with a single (task A, associative learning) or multiple (task A*, cue combination) symbol(s) or cue(s) and then chooses between two actions. The RNN obtains a positive or negative outcome that depends on the probabilistic association between the presented cue(s) and the chosen action. Task environment 2: The RNN is repeatedly presented with a slot machine with two arms to choose from. The RNN receives a reward as a function of the reward probability associated with the chosen arm (task A, fixed; and task A*, reversing within a single game).

and Methods). For comparison purposes, we also considered other regularization mechanisms: explicit regularization by penalizing the L1 norm of recurrent weights or favoring decision entropy in the objective function (*22*, *23*) or implicit regularization through random inactivation of recurrent units [dropout; (*17*)] or additive noise affecting the input to the network (input noise). Note that like dropout and input noise, computation noise corresponds to a form of implicit regularization (no explicit regularization term is added in the loss function). Note also that favoring decision entropy (i.e., "exploration" rather than "exploitation") results in increased readout (policy) noise.

After training, we tested the behavior of $n = 50$ of RNNs trained using increasing levels of computation noise (for other regularization mechanisms, see fig. S1) in a task A*, a variant of the task known as the "weather prediction" task (*24*–*26*), which requires predicting rewarded actions based on sequences of different stimuli seen in task A (Fig. 2A). The behavior observed in task A* can be used to infer what agents (whether human subjects or RNNs) have learnt during task A, from fixed stimulus-action associations that cannot be combined across stimuli to probabilistic associations that can be combined using Bayes' rule to improve the prediction of the rewarded action (Fig. 2B). The weights of RNNs were frozen after training on task A, such that their behavior in task A* could be used to infer what they have learnt during task A.

Although both exact and noisy RNNs performed optimally in task A on which they were trained, their behavior differed markedly in task A* (Fig. 2B). Like RNNs trained without regularization, RNNs trained using explicit regularization mechanisms (including those favoring decision entropy) did not improve their reward rate in task A* with more than a single cue, indicating that they were unable to combine probabilistic stimulus-action associations (fig. S1). By contrast, RNNs trained using implicit regularization mechanisms, either computation noise (Fig. 2B), dropout (fig. S1), or input noise to a smaller extent (fig. S1), made more rewarded actions in task A* with multiple cues than that with a single cue, indicating that they were able to combine probabilistic associations across stimuli in each sequence. Among these mechanisms, RNNs trained with computation noise reached the highest reward rate in task A* (reward rate; dropout with dropout rate = 80%: 0.847 ± 0.001; computation noise with $\sigma = 1.0$: 0.873 ± 0.001; two-sample $t$ test; computation noise versus dropout: $t_{98} = 17.6$, $P < 0.001$). Computation noise yielded a reward rate close to the Bayes-optimal combination of presented cues in task A* (0.919 ± 0.001).

We performed logistic regressions of the behavior of RNNs in task A* (see Methods) to show that the decisions of RNNs with computation noise relied on individual cues as a function of their reliability [Fig. 2C; repeated-measures analysis of variance (ANOVA), main effect of objective reliability on subjective reliability: $F_{3,147} = 758.7$, $P < 0.001$] and reflected the information provided by all cues in the sequence with a moderate "primacy" bias (all logistic weights significant, $P < 0.001$; Fig. 2C). By contrast, exact RNNs made decisions based only on the first cue in the sequence (for other

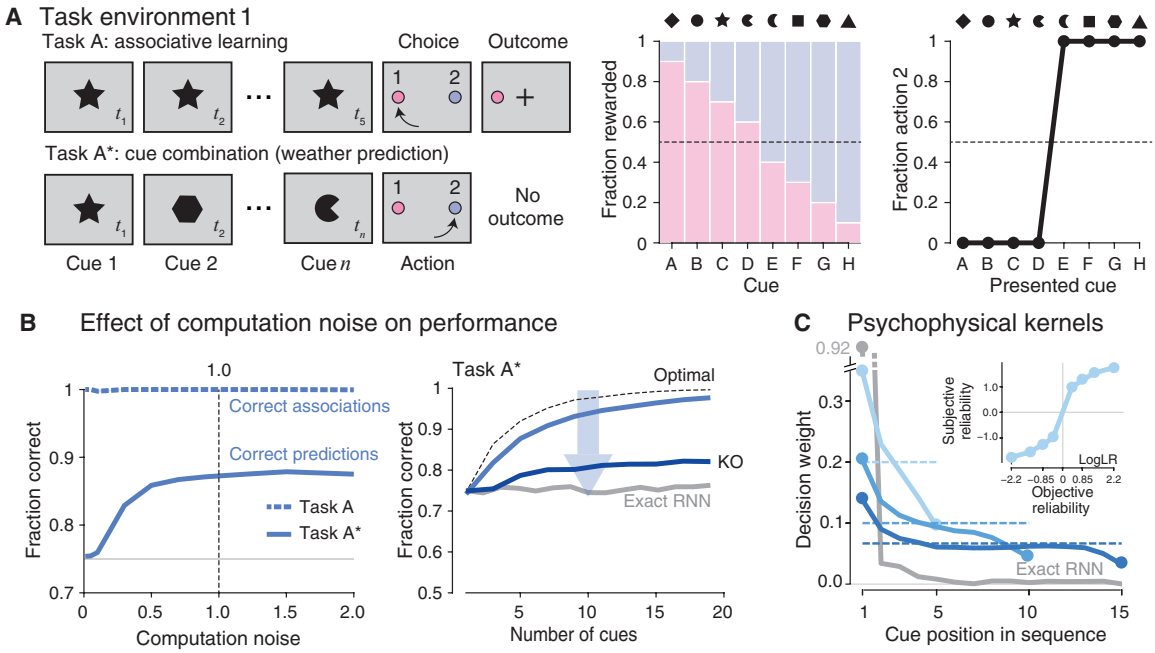


**Fig. 2. Zero-shot performance in the weather prediction task.** (**A**) Description of task environment 1. Left: In task A, the agent is presented with five samples of the same cue among eight possible cues, each of which predicts probabilistically the rewarded action in the current trial. In task A*, the agent predicts the rewarded action based on sequences consisting of samples of different cues. Middle: Fraction of trials for which each action is rewarded for each of the eight cues. Right: Fraction of trials for which action 2 is chosen in response to each cue after training in task A, for exact and noisy RNNs. (**B**) Left: Fraction of correct stimulus-response associations in task A (dashed lines) and fraction of correct predictions in task A* (solid lines) in responses to sequences of five cues, for RNNs trained in task A and tested with increasing amounts of computation noise (*x* axis). The gray line corresponds to the fraction of correct predictions in task A* if only the first cue is taken into account. Right: Fraction of correct predictions in task A* for increasing numbers of cues, for RNNs trained and tested with computation noise (light blue), RNNs trained with computation noise knocked out (KO) during testing (dark blue), and exact RNNs. (**C**) Psychophysical kernels for RNNs with computation noise (blue, $\sigma = 1$) and exact RNNs (gray) for sequences of 5, 10, and 15 cues. Dashed lines indicate flat (ideal) integration kernels. Inset: Relation between the objective (*x* axis) and subjective (*y* axis) reliabilities of individual cues in the decision process. LogLR, log-likelihood ratio.

regularization mechanisms, see fig. S1). The combination of successive cues in each sequence as a function of their individual reliabilities, irrespective of their positions in the sequence, corresponds to the optimal Bayesian inference strategy in the weather prediction task A*.

## Noise-triggered functional regularization in the weather prediction task

If computation noise provides functional regularization to RNNs, then knocking out computation noise after training of the network weights should impair the hallmarks of Bayesian inference identified in RNNs. By contrast, if computation noise only provides structural regularization to RNNs, then knocking it out after training should either improve or not affect accuracy. Knocking out computation noise strongly impaired accuracy during task A* (Fig. 2B), whereas knocking out dropout and input noise improved accuracy during task A* (fig. S1). These opposite effects of knockout for computation noise versus dropout and input noise indicate that computation noise provides both structural and functional regularization, whereas dropout and input noise provide only structural regularization to RNNs during weight training.

To understand how computation noise provides functional regularization to RNNs, we studied their weights (structure) and activity patterns (function) using dimensionality reduction techniques. Because the behavior of noisy RNNs is consistent with Bayesian inference, do their activity patterns represent Bayesian variables? To address this question, we extracted the first two principal components (PCs) of activity patterns in noisy RNNs during task A*. PC1 tracked the posterior belief (log-posterior probability ratio) regarding the rewarded action, whereas PC2 reflected the likelihood (log-likelihood ratio) associated with the current stimulus (Fig. 3A and fig. S2). PC1 did not only show a timescale compatible with Bayesian inference: It followed the time course of the ideal Bayesian posterior belief in individual sequences, even those showing nonmonotonic profiles (fig. S2).

We then examined the input weights associated with each cue, at the input of the recurrent layer (fig. S3). The first PC of input weights in noisy RNNs explained 90% of the variance, compared to 25% for exact RNNs (fig. S3A). This substantial variance explained by a single component suggests that, unlike exact RNNs, noisy RNNs project all eight cues along a single dimension of activity. Furthermore, a cosine similarity analysis revealed that cues associated with the same rewarded action project in the same direction, while those associated with different actions project in opposite directions (fig. S3A). Furthermore, the principal components analysis–based projection of the input weights associated with each cue on the first PC of input weights revealed an ordering of the cues as a function of their reliabilities (fig. S3B).

Last, we studied the statistics of the output activity triggered by different stimuli in noisy RNNs on the decision axis of the networks

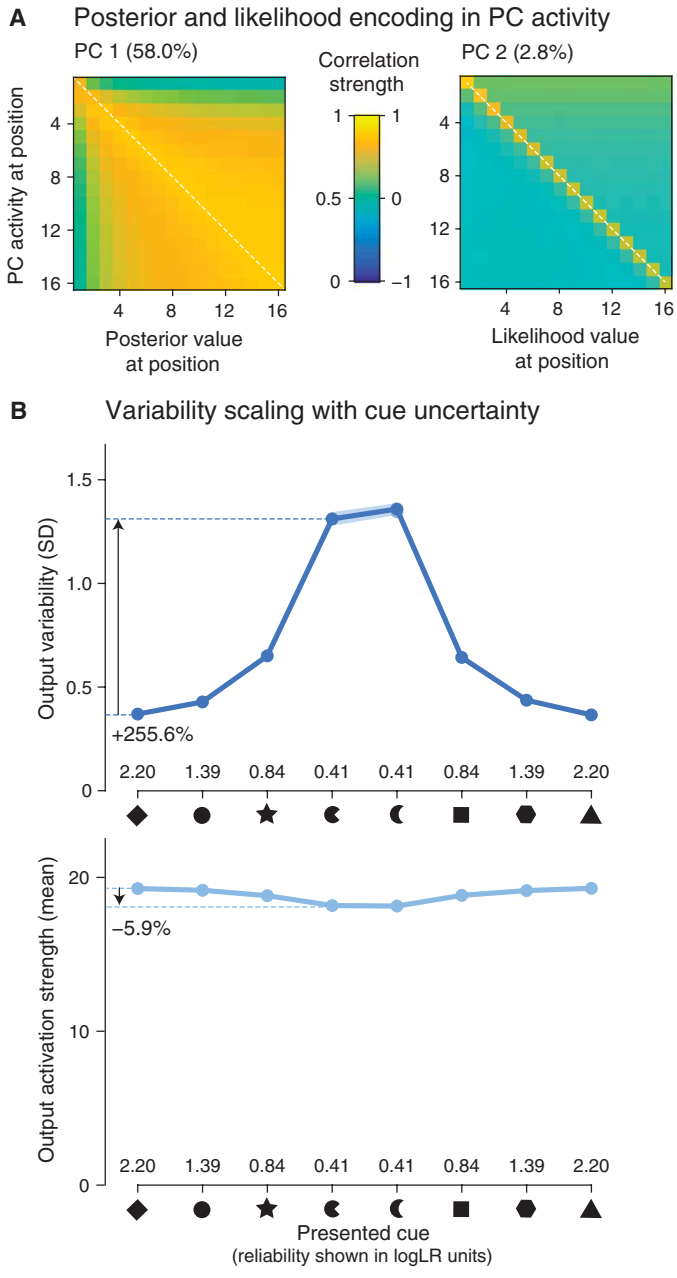


**Fig. 3. Activity patterns in recurrent networks featuring computation noise during the weather prediction task.** (**A**) Temporal cross-correlation matrices for RNNs with computation noise (σ = 1), between the ideal log posterior (*x* axis) and PC1 activity (*y* axis) (left) and between the ideal log-likelihood (*x* axis) and PC2 activity (*y* axis) (right). RNNs with computation noise encode the ideal log posterior and log likelihood with near-zero lag across the sequence. (**B**) Variability scaling with cue uncertainty. Output activity (light line, mean; and dark line, SD) associated with the eight cues for RNNs trained and tested with computation noise. The SD of output activity increased by 255% from the most reliable to the least reliable cues, whereas their mean only decreased by 6%. Reliability is expressed in log-likelihood ratio (logLR) units, i.e., the magnitude of log-likelihood ratio associated with each cue regarding the rewarded action (1 or 2).

(the difference in output activations between action 1 and action 2), at the output of the recurrent layer (Fig. 3B). We found that output variability decreases with stimulus reliability ($F_{3,147}$ = 2349.4, $P < 0.001$), making less reliable stimuli more variable in terms of their activity patterns (+255.6%). By contrast, the magnitude (absolute mean) of output activity on the decision axis of the same noisy networks did not show such a strong relation to stimulus reliability (−5.9%; Fig. 3B). This last observation indicates that RNNs with computation noise represent cue reliability implicitly in the trial-to-trial variability of their decision-relevant activity, not in its magnitude.

## Zero-shot performance in the reversal learning task

Until now, RNNs were trained to learn rewarded stimulus-action associations during task A but did not learn novel associations during task A*. To understand whether computation noise promotes structural or functional regularization during the learning of rewarded actions in uncertain conditions, we trained the weights of $n = 50$ RNNs on a new task A consisting of two-armed bandits with fixed reward schedules (Fig. 4A). As previously, computation noise was modeled as additive normally distributed noise (Fig. 1; see Methods). We then tested the behavior of RNNs in a variant task A* consisting of bandits with volatile reward schedules, an uncertain condition adding unexpected uncertainty (external volatility) compared to that in task A (Fig. 4A). The "baseline" task A includes only expected uncertainty induced by the stochastic nature of presented rewards: The most rewarded action yields a reward with probability $P = 0.95$, whereas the least rewarded action yields a reward with probability $P = 0.05$.

First, we assessed whether RNNs trained on task A could adapt to a reversal in reward contingencies occurring in the middle of an episode in task A*. Although exact and noisy RNNs performed optimally in task A (Fig. 4B, left), only networks with computation noise (Fig. 4B, right) rapidly adapted their behavior following reversals on task A*. Noisy RNNs adapted their behavior much more efficiently than exact RNNs following each reversal (difference in fraction reversed, corresponding to the fraction of actions toward the most rewarded arm in the second half of task A*, after the reversal: $t_{98} = 8.7$, $P < 0.001$; for other regularization mechanisms, see fig. S4). Knocking out computation noise after training in noisy RNNs significantly slowed down reversal dynamics, measured in terms of their reversal time constants (see Methods; Fig. 4C).

These behavioral differences between noisy and exact RNNs were investigated in terms of their activity patterns using dimensionality reduction techniques (Fig. 4D). We decoded (using a linear decoder) the previous reward (positive or negative at time $t - 1$) and the current action (action 1 or 2 at time $t$) based on network activity in the recurrent layer at time $t$ (see Methods) and then projected the network activity onto these two decoded axes (*x* axis, current action activity; and *y* axis, previous reward activity). For noisy RNNs, we found that the "change of mind" of the network represented by the "next action" dimension followed a change of sign in the representation of previous reward (from positive to negative). By contrast, exact RNNs were unable to switch away efficiently from their initially preferred action, despite the fact that their representation of the previous reward changed sign after the change, as in noisy RNNs.

## Hallmarks of meta-learning in noisy neural networks

As observed in human learning, we next hypothesized that noisy RNNs could adapt their behavior to volatile reward schedules. We formalized the question by asking if they adapted their learning rate parameters to unexpected uncertainty (volatility). This form of "meta-learning," a characteristic of human learning in volatile

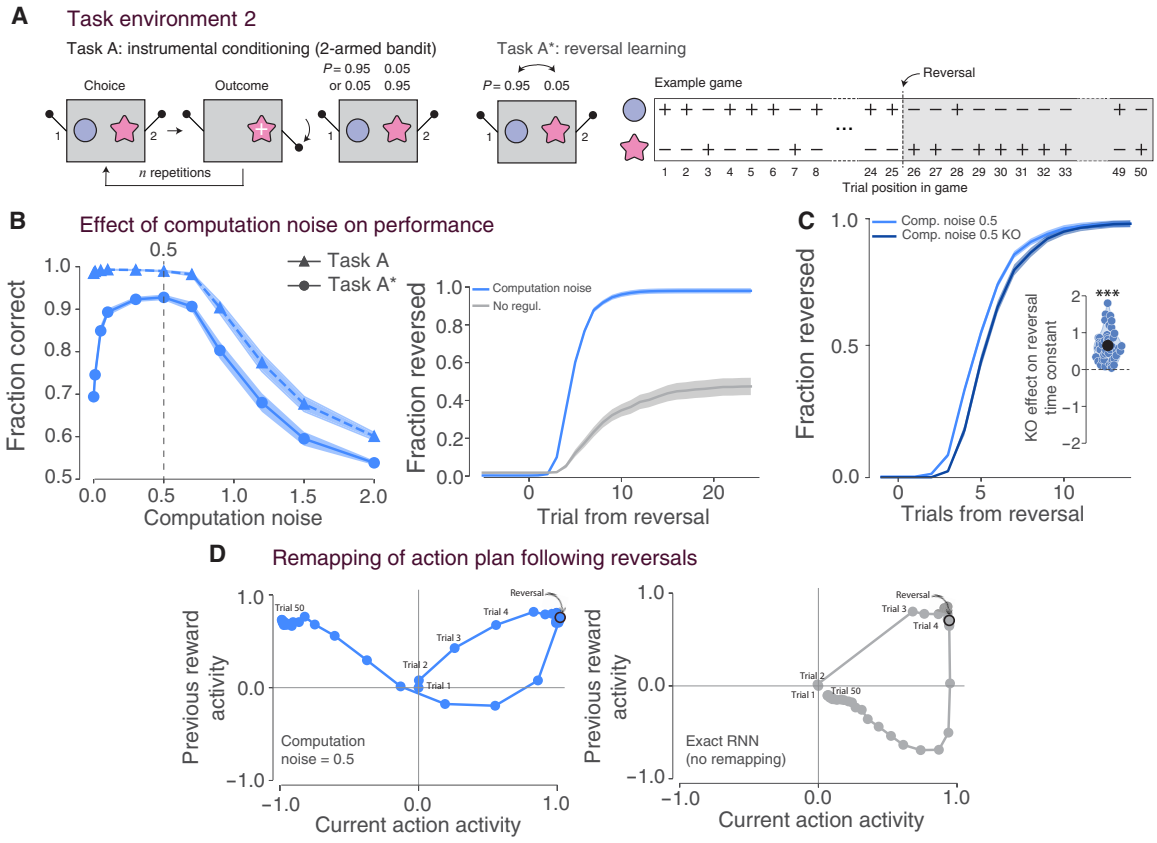


**Fig. 4. Zero-shot performance in the reversal learning task.** (**A**) Description of task environment 2. Left: Description of task A with fixed reward schedules. The most rewarded arm varies randomly across games such that the agent needs to learn which arm is most rewarded in each game. Right: Description of the reversal learning task A*. The reward probabilities associated with the two arms reverse in the middle of the game, such that the agent needs to switch away from a previously reinforced action. (**B**) Effect of computation noise on performance. Left: Performance (proportion correct) as a function of the regularization parameter for different RNN types on train task A and test task A*. RNNs with moderate levels of computation noise ($\sigma \sim 0.5$) adapt efficiently their behavior to the reversal. Exact RNNs correspond to computation noise = 0 (leftmost point on *x* axis). Right: Reversal curves for noisy RNNs (blue, $\sigma = 0.5$) and exact RNNs (gray). (**C**) Reversal curves of RNNs trained with computation noise either present (light blue) or knocked out (KO; dark blue) during testing. Inset: KO effect on reversal time constant. Knocking out computation noise leads to slower reversals in response to changes in reward probabilities. (**D**) Mean trajectories of activity patterns in the two-dimensional space predicting the action plan and the previous reward from recurrent activity (in arbitrary units). ***$P < 0.001$.

environments in the absence of explicit instructions (*27*), has previously been implemented either by explicit sophistication of process-based models (*28*) or by explicitly training the weights of RNNs on volatile reward schedules (*23*). We compared the performance of RNNs trained on task A on a new task A* consisting of two conditions tested separately: a first "stable" condition using fixed reward schedules and a second "volatile" condition with multiple reversals in reward schedules (Fig. 5A). Both conditions differed from the training task A. The stable condition exhibited increased expected uncertainty (due to an increase in reward stochasticity, from 0.05 to 0.20). The volatile condition had equal expected uncertainty but included an additional source of unexpected uncertainty, with reversals in reward contingencies occurring every 25 trials.

In agreement with the ability of RNNs to change their behavior in response to a single reversal (Fig. 4), we found that noisy RNNs outperformed exact RNNs in the volatile condition without any training in this condition (Fig. 5). These networks not only performed better but also adapted their learning rate in this condition (Fig. 5; for other regularization mechanisms, see fig. S5). We also tested the RNNs on the volatile bandit task after training them on that same task A* (fig. S6). In agreement with previous established results, we found that exact and noisy RNNs performed virtually as well and both exhibited an adaptation of their learning rate to changes in reward schedule volatility. Crucially, unlike what is observed for RNNs trained in stable conditions, knocking out computation noise for RNNs trained and tested in the same task conditions (i.e., without generalization) led to increased task performance. This last finding shows that computation noise does not provide functional regularization for RNNs when the networks are not evaluated in their generalization abilities (i.e., tested in a task condition that was not used for training the networks).

Together, these results show that the accounts of meta-learning that involve complex Bayesian inferences or sophisticated RNNs trained in volatile conditions are sufficient but not necessary. Moderate levels of computation noise provide the same capabilities with zero training in conditions involving volatile reward schedules.

We observed that noisy RNNs consistently outperformed other regularized RNNs in the volatile condition, which was not seen during training (fig. S5). We hypothesized that this advantage of computation noise comes from a "functional" regularization enabled by a relevant emergent population-level structure of computation noise.

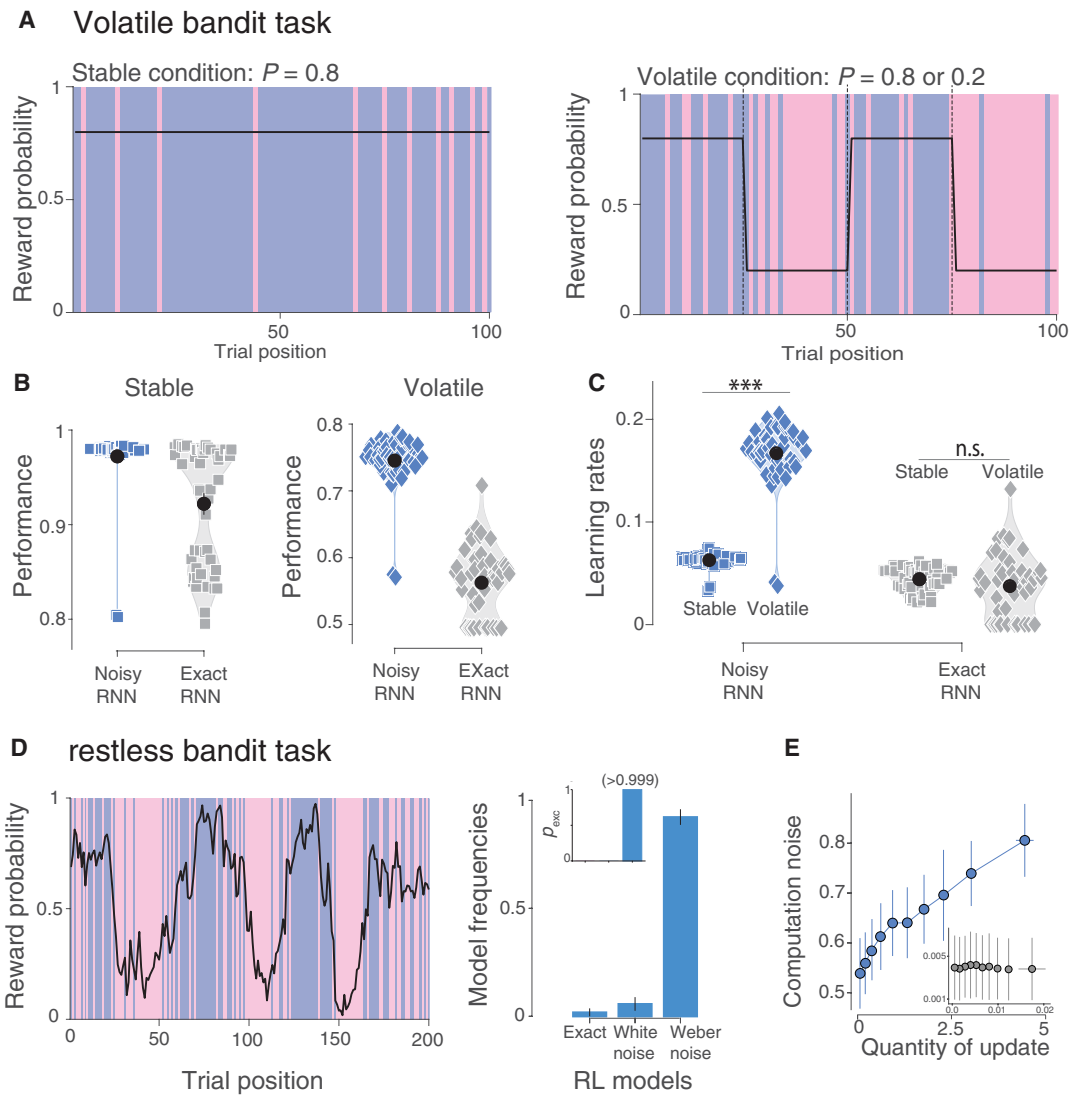


**Fig. 5. Adaptation to volatile schedules and emergent noise structure.** (**A**) Description of the volatile bandit task A*. Left: Stable condition with fixed reward probabilities. Right: Volatile condition with reversing reward probabilities every 25 trials. (**B**) Performance correct achieved by RNNs with (blue, σ = 0.5, left) and without (gray, right) computation noise in the stable (left) and volatile (right) conditions. RNNs with computation noise substantially outperform exact RNNs in the volatile condition. (**C**) Best-fitting learning rates for the different types of RNNs in the stable and volatile conditions. Unlike RNNs without computation noise, RNNs with computation noise exhibit an adaptation of their learning to the volatility. (**D**) Description of the restless bandit task A*. Left: The reward probabilities associated with the two arms drift randomly over the course of the game (200 trials). Right: Bayesian model comparison between Q-learning RL models with no computation noise, white computation noise and Weber-structured computation noise. The Q-learning RL models were fitted on simulated actions from the RNNs with computation noise (σ = 0.5). The behavior of RNNs with computation noise is better fitted by a Q-learning RL model featuring a Weber-like noise structure. (**E**) Relationship between the quantity of update in the RNNs with computation noise and the noise corrupting the update. To obtain the two dimensions, we projected the quantity of update and the computation noise in the recurrent activity on the decision axis. As predicted by a Weber-like noise structure, the population-level computation noise scales with the quantity of update in the network. Inset: Same relationship for RNNs trained without computation noise but tested with computation noise. ***$P < 0.001$; n.s., nonsignificant.

In human reward-guided learning, it has been established that the amount of internal noise corrupting each update of action values scales with the amount of update itself (*7*, *8*). This Weber-like structure is relevant for learning in volatile conditions, as it enables efficient exploration following unexpected outcomes. When the state of the environment becomes uncertain, prediction errors (PEs) and value updates become large, which result in increased internal noise. Here, although each recurrent unit is corrupted by independent sources of noise, we hypothesized that the Weber structure observed in humans could emerge at the population level in noisy RNNs.

Following recent work (*7*), we presented the RNNs trained on task A with "restless" bandits whose reward probabilities drift continuously over time (Fig. 5D and fig. S7 for restless bandits with continuous drifting rewards). We then fitted Q-learning RL models, which track expected rewards, to their behavior. We considered three RL model variants: The first assumes that value updates are performed exactly (variant #1), and the two others assume that

value updates are corrupted by white (variant #2) and Weber noise (variant #3). On par with previously established results in humans, we found that the behavior of noisy RNNs was better explained by a noisy RL model featuring Weber noise (Fig. 5D). This result means that the Weber structure of computation noise observed in human learning naturally emerges at a population level in noisy RNNs. To validate this finding, we estimated and observed a monotonically increasing relation between the quantity of updates in the recurrent dynamics of noisy RNNs and the associated computation noise at the population level (Fig. 5E). This last result confirms the emergent Weber structure of computation noise in a way that is agnostic to the particular algorithmic formulation of learning that was used in Fig. 5D to fit the behavior of noisy RNNs. This result validates this Weber noise structure without assuming that the RNN conforms to the Q-learning RL model, by showing that the variability of updates in the recurrent layer scales with the magnitude of updates, in a way that is agnostic to the specific cognitive model of learning that is instantiated by the RNN.

### Noise-triggered functional regularization in the reversal learning task

This emerging Weber structure leads to interesting attractor destabilizations. When changes in the environment occur, negative rewards are observed, inducing high PEs and large updates. Destabilizing the internal representation proportionally to the amount of updates should enable a faster relearning of the new contingencies after the environment switch. We validated that computation noise bears this role of functional regularization by disabling it after weight training. As predicted, knocking out computation noise after training reduces the speed of adaptation after an environment change (Fig. 4C). This causal perturbation confirms that computation noise, unlike dropout (see fig. S5), acts as a functional regularizer and is instrumental to zero-shot adaptation to unexpected uncertainty in the reversal learning task.

Decision-making models of human learning have implemented computation noise by corrupting otherwise deterministic (and often optimal or near-optimal) latent variables (*3*, *7*, *8*). By doing so, these noisy models predict a monotonic decrease of task performance with the level of computation noise. By contrast, noisy RNNs show a non-monotonic, inverted U-shaped relation between the level of computation noise and task performance, whereby maximal performance on A* is obtained for moderate levels of computation noise (Fig. 4B). To test this specific prediction, we related the amount of computation noise in a group of $n = 198$ human participants to their performance in a restless bandit task tested online (Fig. 6; see Methods). Like noisy RNNs, but unlike the basic predictions of noisy RL models tested in the same conditions, we found that human participants with the lowest levels of computation noise performed less accurately than participants with moderate levels of imprecisions (Fig. 6C). This non-monotonic relation provides a first piece of empirical evidence that computation noise promotes the same kind of functional regularization in humans and noisy RNNs.

In addition, we found that the optimal levels of computation noise in noisy RNNs were compatible with those observed in human participants. Specifically, the best-fitting Weber fractions for noisy RNNs were found to be 0.20 [0.18, 0.23] (median [first quartile, third quartile]), in line with those of the 29 human participants from (*7*) tested in the laboratory (0.25 [0.19, 0.34]), and those of the 198 participants tested online (0.37 [0.21, 0.52]). In this second dataset, we found the Weber fractions of noisy RNNs to exhibit stronger alignment to the top-performing 50 participants (0.27 [0.19, 0.45]). Note that we do not expect perfect alignment between best-fitting Weber fractions as we have previously shown (*7*) that around 30% of the computation noise fitted from human behavior is attributable to deterministic biases and not genuine variance in underlying computations.

## DISCUSSION

In cognitive psychology, general intelligence is not measured by the level of skill (or expertise) at any given task: Recent definitions describe general intelligence as skill acquisition rather than skill itself (*28*, *29*). In other words, intelligent agents are not necessarily the

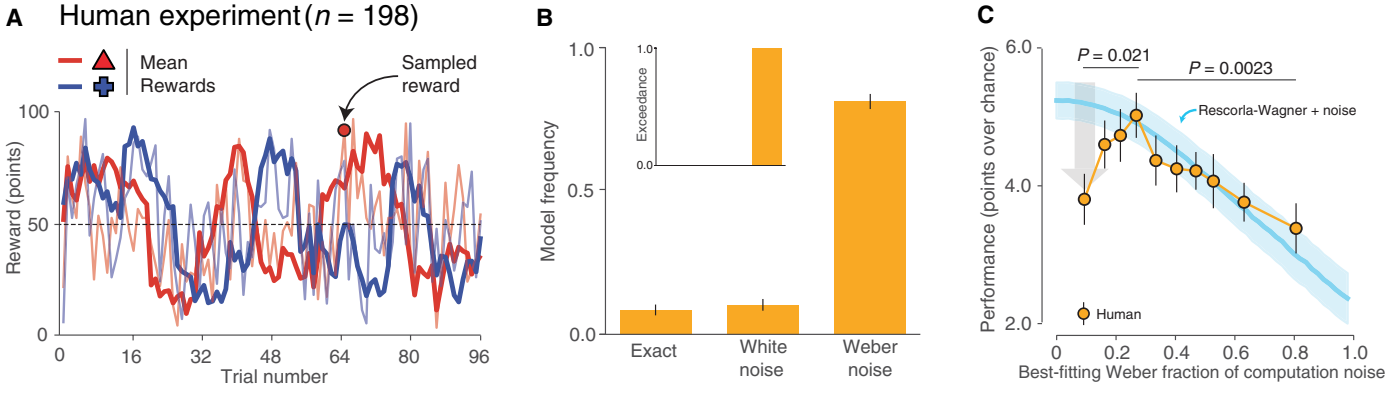


**Fig. 6. Moderate levels of computation noise improve performance in human participants.** (**A**) Description of the experimental paradigm. Example of drifts in the magnitude of rewards that can be obtained from the two arms. Rewards were sampled from probability distributions with means that drifted independently across trials. Thick lines represent the drifting means of the two probability distributions, whereas thin lines correspond to reward samples drawn from the probability distributions that can be obtained if chosen in each trial. (**B**) Bayesian model selection results for $n = 198$ human participants. On par with computation noise RNNs and previous published results, humans were best explained by a Q-learning RL model corrupted with Weber-structured noise at each update step. (**C**) Excess points earnt compared to chance level by human participants ($n = 198$ in yellow, binned in deciles) and a noisy RL algorithm corrupted by Weber-structure additive noise (in cyan). If computation noise only impeded human learning, then human performance would decrease with the level of computation noise, similarly to the noisy RL algorithm. Moderate levels of computation noise improve performance in tested human participants. Error bars and shaded areas represent SEM.

most skilled at any given task, but they are able to acquire new skills from previously acquired skills at low (ideally, zero) cost. Our approach for measuring zero-shot adaptation to uncertainty consists in training agents to perform a first task A and then testing their ability to perform a more challenging variant A* of the same task that requires taking into account a source of uncertainty not necessary to solve task A. Teaching humans and other animals to perform a challenging task typically requires training on the different subcomponents of the task. We reasoned that an intelligent agent trained on task A should be able to behave adaptively in task A* without additional training, a form of acquisition of cognitive ability during training on task A that becomes expressed when tested on task A*.

Understanding the origin of adaptive behavior in uncertain conditions constitutes an important challenge for neuroscience research, and recent efforts typically proceed by engineering cognitive abilities through purposeful sophistication of neural architectures (*23*, *30*). For example, Bayesian theories of brain function propose specific coding mechanisms to decode external stimuli from noisy sensory input (*15*, *31*). Similarly, the adaptation of sensory- and reward-guided learning to volatile conditions has been modeled explicitly through hierarchical inference (*27*, *32*) or hierarchical RL (*33*, *34*) in neural circuits, a form of explicit sophistication of process-based models, or by explicitly training the weights of RNNs on volatile reward schedules (*23*). Here, we have explored a different avenue, by hypothesizing that the adaptation to uncertainty during learning and decision-making may be supported, at least, in part, by computation noise, a pervasive property of biological neural networks. Other relevant work has recently shown that the adaptation of learning rates to volatility can also be triggered by simpler learning algorithms that do not involve computation noise, such as specific forms of leaky integration (*35*) and gated recurrence at the level of individual units in RNNs (*36*), where networks with gated recurrence trained at specific nonzero levels of volatility are able to adapt their learning rates after training to other levels of volatility.

The substantial benefits of computation noise described above call to reconsider the status of the ubiquitous internal variability of neural activity. Input noise (e.g., sensory noise) is widely seen as a hard constraint that neural networks have to cope with using specific mechanisms: population coding to average out input noise or efficient coding to limit its effects on the decoding of sensory signals (*9*–*12*). These mechanisms typically do not consider noise arising from neural computations themselves, beyond input noise. We see computation noise, like input noise in previous work, as a biological constraint on neural activity, present both during and after training of network weights. Our findings do not depend on strong assumptions regarding the origin of computation noise. It may reflect not only genuine biophysical stochasticity (*1*) but also background, task-irrelevant activity (*37*) that may not be random in an absolute sense but, nevertheless, triggers substantial variability in decision signals (*38*).

Noisy RNNs share similarities with variational autoencoders (VAEs) (*39*), particularly in the positive impact of noise in the latent state on robustness to adverse (e.g., uncertain) conditions. However, in VAEs, the level of noise is itself the output of a neural network, and the structure of variability in VAEs is thus dynamically learnt through backpropagation to minimize the loss. Similar approaches have been used in RL (*40*, *41*) by introducing noise in network dynamics whose parameters were learnt through backpropagation. After training of its parameters, noise promoted efficient exploration that outperformed standard "epsilon-greedy" policies. By contrast, the level of computation noise in noisy RNNs is preset and fixed during training, making noise a functional constraint rather than an adjustable parameter.

Policy noise, often referred to as readout noise, is another frequently considered source of noise in neural networks. This type of noise is commonly introduced to enhance exploratory behaviors during training. To examine the influence of policy noise on the network's ability to adapt to uncertainty, we incorporated the entropy of the readout policy in the loss function, which specifically promotes policy noise (see Methods). Our results indicated that, unlike computation noise, policy noise did not facilitate the network's adaptation to the added source of uncertainty encountered in task A* (see figs. S1 and S4).

By corrupting the recurrent information, computation noise forces the RNN to integrate information across time (or over cues), which makes it more flexible and adaptable to variability in input (different symbols presented sequentially in the weather prediction task and changes in the reward probabilities in the bandit task). This adaptability arises from a specific functional form of regularization, distinct from "structural" regularizations (either explicit methods such as the L1 norm of recurrent weights or the entropy of the policy in the loss function, or implicit methods such as dropout and input noise). Moreover, computation noise induces less intuitive effects. One such effect is a form of meta-learning conferred by computation noise in response to changes in the volatility of reward schedules (*27*). This behavior is not replicated in exact (noise-free) RNNs unless they are trained and tested under identical task conditions, demonstrating the unique influence of computation noise. A second effect corresponds to the scaling of the effective noise on the decision dimension with input reliability, increasing noise in response to less reliable symbols in the weather prediction task, and increasing noise in response to unexpected outcomes in the bandit task.

Our findings emphasize the distinction between structural and functional regularization in neural networks. Structural regularization corresponds to the tuning of connection weights during training, and, in this regard, computation noise produces similar benefits as heuristics commonly used in the literature such as dropout (*17*) or explicit cost terms in the objective function (*22*, *23*). By contrast, functional regularization corresponds to the shaping of stochastic activity dynamics after training, and computation noise provides unique benefits in this regard. First, computation noise allows transitions between otherwise stable attractors. Noisy networks can integrate the information provided by successive stimuli in the weather prediction task, whereas exact networks consider only the first (or last) presented stimulus. In the bandit task, noisy networks can change their behavior in response to changes in reward contingencies, whereas exact networks are essentially blind to changes in reward contingencies after the first few trials. Second, computation noise allows the reliability-dependent weighting of stimulus information in the weather prediction task and the regulation of reward learning rates as a function of volatility in the reversal learning task. These benefits of computation noise are hallmarks of Bayesian and human inference (*27*, *42*), only present in exact networks that have been trained on conditions that require these two features to achieve high accuracy.

Structural and functional forms of regularization are not independent of each other, because computation noise acquires a specific structure during training (on task A) that makes it particularly efficient for adapting to uncertainty after training (on task A*). After

the learning of probabilistic associations between individual stimuli and rewarded actions, computation noise scales with the unreliability of these associations to provide reliability-dependent weighting of these associations. In addition, after learning to play two-armed bandits, computation noise scales with the PE associated with obtained rewards to regulate reward learning rates as a function of volatility. Critically, adding computation noise (zero-mean normally distribution noise with fixed SD) to RNNs after training does not result in the same uncertainty-scaling structure (Fig. 5E, inset), indicating that weight tuning (the learning of the network weights through backpropagration) actively shapes the structure of computation noise.

In both cases, weight tuning shapes computation noise to minimize the loss of reward-predictive information due to noise. The backpropagation process could adjust the weights in such a way that noise is averaged out along the decision axis of the RNN. This reduction of computational noise on the decision axis is particularly crucial for maximizing rewards when the input holds substantial information. When the presented cue is highly predictive of the rewarded action or when one of the two bandit arms is associated with a high reward probability (and thus small reward PEs), weight tuning should decrease random variability on the decision axis to maximize accuracy. Conversely, when the presented cue is weakly predictive of the rewarded action or when the two bandit arms are associated with uncertain outcomes (and large reward PEs), weight tuning does not require decreasing random variability on the decision axis to maximize accuracy and may even leverage computation noise to generate exploratory behavior (*3*). Nevertheless, further research is necessary to precisely identify the mechanisms that generate this uncertainty-scaling, Weber-like structure of computation noise, which is consistent with recent behavioral observations in humans (*7*, *8*). We have used in this study a tanh, symmetrical nonlinearity on the activation of RNN units, but future work should explore other forms of nonlinearity such as ReLU that would, unlike the sigmoidal nonlinearity that we have used, retain noise in the linear part of the function. Future research should also explore the impact of correlated noise across units or heterogeneity in the level of noise across units on noisy RNNs, particularly how noisy RNNs may exploit this heterogeneity or correlation in noise levels across units, a setting that may reflect better the variability of the activation of biological neural networks.

Existing approaches to dealing with uncertainty in neural networks proceed by training neural networks in the same uncertain conditions where they are subsequently tested (*23*). The training of the structural connectivity of artificial neural networks is thought to reflect long-term (genetic and developmental) influences (*43*, *44*). The fact that noisy networks can adapt in a zero-shot fashion to uncertain conditions that they have never experienced during training (sequences of multiple predictive stimuli in the weather prediction task or changing reward contingencies in the reversal learning task) suggests that weight tuning may primarily learn stable structural properties of our environments (*30*) rather than specific, context-dependent forms of uncertainty, which differ even between the weather prediction task (expected uncertainty) and the reversal learning task (unexpected uncertainty). By shaping computation noise to scale with task uncertainty, weight tuning may have evolved to leverage computation noise to generate adaptive behavior in uncertain environments without requiring extensive training in each of them. This is particularly efficient, given that weight tuning itself is typically less efficient and slower in uncertain environments, as can be seen in the context of learning stimulus-response associations by training RNNs on sequences of cues instead of training them on single cues (see fig. S8).

The fact that computation noise promotes the same zero-shot adaptation to different forms of uncertainty (expected uncertainty in the weather prediction task and unexpected uncertainty in the reversal learning task) is particularly compelling. Across the two experimental frameworks, similar moderate levels of computation noise optimize performance in task A* after training on task A, and this optimal level of computation noise is compatible with behavioral measurements in humans (*6*, *7*). At the neural level, the activity patterns of noisy neural networks in the weather prediction task, particularly their low-dimensional trajectories, are highly consistent with neural observations from the lateral intraparietal cortex of macaque monkeys engaged in the weather prediction task (*26*, *37*). In the bandit task, the dissociation between the coding dimensions of action values and action outcomes found in noisy neural networks is also compatible with multimodal neural observations from the medial prefrontal cortex (*45*–*47*). Recent findings in the literature suggest that the benefits of computation noise extend beyond learning and decision-making problems. For example, the introduction of cortical-like stochastic noise in RNNs trained to perform sensory inferences confers specific properties such as divisive normalization of network activity and stimulus-modulated noise variability (*48*). Similarly, training deep convolutional neural networks (CNNs) to perform image recognition with a first layer whose units exhibit key properties of cells in the primary visual cortex (V1), including stochastic noise, substantially improves the network robustness to adversarial image perturbations through both structural and functional regularization (*49*), exactly as it is the case of RNNs in the tasks that we have tested in this study.

Other regularization mechanisms did not provide the same cognitive benefits as computation noise in task A*. Explicit weight regularization (*50*) did not result in adaptive behavior in the weather prediction task, and explicit entropy regularization (*22*, *23*) did not provide any benefit in task A* after training on task A, across both experimental frameworks. Dropout (*17*) substantially improved performance in task A*, but widely different fractions of randomly inactivated units were necessary to improve performance in the weather prediction task (80%) and in the reversal learning task (50%). Furthermore, and unlike computation noise, dropout provided purely structural regularization: Performance in task A* increased when dropout was turned off after training, whereas performance in task A* decreased when computation noise was suppressed after training.

Despite its unexpected benefits, we do not mean that computation noise is sufficient to explain the several complex forms of generalization observed in humans and other animals, beyond the zero-shot adaptability to uncertainty that we studied here (*26*, *51*). The difference between task A and its variant A* was carefully chosen to study the adaptability of neural networks to a form of uncertainty not required to solve the task on which they were trained. This is the same for the benefits of stochasticity in the V1-like layer of deep CNNs trained to perform image recognition and tested in adversarial conditions (*49*). Nevertheless, we believe that the benefits of computation noise are compelling, especially because they are shared across tasks which are qualitatively different regarding the status of the decision-maker. In the weather prediction task, the decision-maker can be described as an observer of presented

symbols and can only predict the correct action to perform based on observed symbols. By contrast, in the two-armed bandit task, the decision-maker is truly an agent that interacts sequentially with the bandit, by controlling which lever to pull on each trial and obtaining the corresponding reward. It is, therefore, not trivial that computation noise confers very similar benefits across these two cognitive tasks that are typically studied by different subfields of decision-making research and modeled using very different types of computational models (Bayesian inference–based models for the weather prediction task and RL-based models for the bandit task).

Furthermore, the controlled laboratory experiments that we have studied here (because they have also been extensively studied in humans) require little to no complex exploration. Therefore, we do not argue that computation noise alone enables the efficient foraging of high-dimensional environments and can replace, for example, dedicated exploration or curiosity (*52*–*54*). To further investigate how computation noise may interact with such strategies for behaving in conditions involving uncertainty, future research could follow recently developed approaches for fitting parameters of RNNs (including their level of computation noise) to human behavior in the cognitive tasks studied here (*55*, *56*). In particular, the idea of developing noisy RNNs that can regulate their level of computation noise as a function of task conditions is appealing, and it is consistent with very recent work showing that humans can increase and decrease their levels of RL noise as a function of the dominant source of uncertainty in a two-armed bandit task (*57*).

To conclude, the benefits of computation noise for cognition under uncertainty move beyond the traditional distinction between signal and noise in information processing systems. Computation noise likely reflects a genuine constraint on neural information processing that is actively shaped at both long (structural regularization of network weights) and short (functional regularization of network activity) terms. However, our findings reveal that intelligence may ride on moderate levels of computation noise to promote efficient behavior when confronted with uncertain environments without any training nor ad hoc top-down sophistication. Testing this hypothesis further in humans and artificial intelligence appears like a promising yet almost unexplored (*40*, *41*) avenue for future research.

## METHODS

### Human participants

Participants ($n = 230$) played a two-armed restless bandit task (139 females; age, $34 \pm 10.2$ years). The experiment was performed on the Prolific platform (prolific.co), and the research was carried out following the principles and guidelines for experiments including human participants provided in the declaration of Helsinki and approved by the relevant authorities (Inserm Ethical Review Committee, IRB00003888). Participants provided a written informed consent before their inclusion. To sustain motivation throughout the experiment, participants could obtain a monetary bonus depending on the number of points won in the experiment.

### Experimental procedures for the bandit task

Participants played two blocks of 72 trials of a canonical restless two-armed bandit task. On each trial, participants observed and received the reward associated with the chosen arm and did not observe or receive the reward associated with the unchosen arm. The rewards observed by the participants (between 1 and 99 points) were sampled from bell-shaped beta distributions whose means followed a random walk process. Because of low performance (more than 2 SDs below the mean), $n = 32$ participants were excluded from analyses.

### Neural network architecture

The artificial neural networks used for the weather prediction task and the bandit task are identical and correspond to standard (Elman) RNNs. Let us call $X_t$ the input to the network, $Z_t$ the recurrent state of the network, and $Y_t$ the output of the network. The RNN is governed by the following equations

$$\hat{Z}_t = \boldsymbol{W} \cdot Z_{t-1} + \boldsymbol{U} \cdot X_t + \boldsymbol{B} \tag{1}$$

$$Z_t = h_Z\left(\hat{Z}_t\right) \tag{2}$$

$$Y_t = h_Y\left(\boldsymbol{V} \cdot Z_t\right) \tag{3}$$

where $h_Z$ is the hyperbolic tangent, $h_Y$ is the softmax (sigmoid) function, and $\langle \cdot \rangle$ is the matrix multiplication operator. $\boldsymbol{W}$, $\boldsymbol{U}$, $\boldsymbol{B}$, and $\boldsymbol{V}$ are four matrices of network parameters adjusted during training. For the weather prediction task, $X_t$ is a "one-hot" vector encoding the presented cue (among eight possible cues). For the bandit task, $X_t$ is composed of the previous observed reward and a one-hot vector encoding the previous chosen action (among two possible actions). Following the first equation, the input $X_t$ and the previous recurrent activity $Z_{t-1}$ are integrated into an updated state $\hat{Z}_t$ through matrix multiplications with weight matrices $\boldsymbol{U}$ and $\boldsymbol{W}$ (plus an additive bias term $\boldsymbol{B}$). This updated state $\hat{Z}_t$ is then passed through a nonlinearity $h_Z$ to give the updated recurrent activity $Z_t$. This updated recurrent activity $Z_t$ then projects to action probabilities $Y_t$ through matrix multiplication with output weights $\boldsymbol{V}$, followed by the softmax $h_Y$ operator. For both tasks, we used $K = 48$ units in the recurrent layer, resulting in 2832 free parameters to adjust during training in the weather prediction task and 2592 free parameters in the bandit task.

### Objective functions

In both tasks, the objective functions used for training the networks are derived from obtained rewards. In the weather prediction, where all task A trials are independent from one another, the objective function is written as follows

$$L(\pi) = \mathbb{E}^{\pi}\left[r \middle| s_{1:N}\right] \tag{4}$$

where $s_{1:N}$ is the $N$ presented cues ($N = 5$), $r$ is the obtained reward, and $\pi$ is the "decision policy" giving the probability of each action (i.e., the output layer of the neural network). In the bandit task where the successive trials are dependent, the objective function is written as follows

$$L(\pi) = \mathbb{E}^{\pi}\left[\sum_{t\geq 1}\sum_{k\geq 0}\gamma^k \cdot r_{t+k}\right] \tag{5}$$

where $t$ is the trial number, $\gamma \in [0,1]$ is the discount factor, $r_{t+k}$ is the obtained reward at trial $t + k$ (where $k$ is a positive integer), and $\pi$ is the decision policy of the neural network. Having no prior assumptions regarding $\gamma$, we used $\gamma = 0.5$.

### Training procedure

The RNNs have a set of parameters (the matrices $\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{W}$, and $\boldsymbol{B}$ described in the "Neural network architecture" section) that we trained using the REINFORCE algorithm. This training procedure relies on a direct differentiation of the objective functions. In the weather prediction task, the gradient of the objective function is written as follows

$$\nabla L(\pi) = \mathbb{E}^{\pi}\left[\nabla \log \pi\left(a \middle| s_{1:N}\right) \cdot r\right] \tag{6}$$

where $s_{1:N}$ is the presented cues, $r$ is the obtained reward, $\pi$ is the decision policy of the neural network, and $a$ is the chosen action. In the bandit task, the gradient of the objective function is written as follows

$$\nabla L(\pi) = \mathbb{E}^{\pi}\left[\sum_{t\geq 1} \nabla \log \pi\left(a_t\right) \cdot \left(\sum_{k\geq 0} \gamma^k \cdot r_{t+k}\right)\right] \tag{7}$$

where $t$ is the trial number, $a_t$ is the chosen action at trial $t$, $r_{t+k}$ is the observed reward at time $t+k$ (where $k$ is a positive integer), $\pi$ is the decision policy of the neural network, and $\gamma$ is the discount factor.

On both tasks, we trained 50 artificial agents using the same training procedure. The behavior and activity patterns of the 50 trained agents were entered as repeated measures in all analyses reported below. The stochastic gradient ascent procedure was performed with the RMSProp optimizer and a learning rate of 0.0001. We set the total number of training steps to 50,000 for the weather prediction task, with each step consisting of 100 independent trials. We also set it to 50,000 for the bandit task, with each step consisting of one game of 100 trials. Asymptotic performance was reached at the end of the optimization procedure in both cases.

### Introducing decision entropy in neural networks

RNNs with decision entropy feature a decision entropy term added to the objective function $L(\pi)$

$$L(\pi) \leftarrow L(\pi) + \eta \cdot S(\pi) \tag{8}$$

where $S(\cdot)$ is the entropy function, $\pi$ is the decision policy, and $\eta$ is a positive scaling factor. This decision entropy term, commonly used for training deep RL networks, encourages explicitly decision policies with high entropies.

### Introducing dropout in neural networks

Dropout RNNs were set to 0 the activity of recurrent units with probability $p_{\text{discard}}$

$$p\left(Z_t^k = 0\right) = p_{\text{discard}} \tag{9}$$

where $Z_t^k$ is the activity of recurrent unit $k$ at time $t$. This defines a "mask," a matrix of zeros and ones, with which we multiply the recurrent activity. We assume that the mask changes across different "runs" but does not change during one "run," a run being defined as one trial for the probabilistic reasoning and one game for the reward-guided learning task. When dropout is knocked out, we scale the activity such that it has the same expected value by multiplying the activity by $p_{\text{discard}}$.

### Introducing L1 penalization in neural networks

RNNs with L1 penalization feature a term added to the objective function $L(\pi)$, which penalizes recurrent weights $W_{\text{rec}}$ by encouraging sparsity

$$L(\pi) \leftarrow L(\pi) + \kappa \mid W_{\text{rec}} \mid \tag{10}$$

where $|.\ \ |$ is the L1 norm, $\pi$ is the decision policy, and $\kappa$ is a positive scaling factor.

### Introducing input noise in neural networks

RNNs with input noise feature normally distributed noise in the sensory inputs

$$X_t \leftarrow \mathcal{N}\left(X_t, \sigma \cdot I_d\right) \tag{11}$$

where $\mathcal{N}$ is the normal distribution, $\sigma$ is a positive scaling factor, and $I_d$ the identity matrix of dimension equal to the number of sensory inputs. In other words, white and independent Gaussian noise is added to each sensory input. In the case of reward-guided learning, the input noise is only added to the previous reward and not to the one-hot encoding of the previous action, as this latter one is not a sensory input (note that even when considering input noise on the previous action, the performance on A* were not increased). This noisy input $X_t$ is then fed to the RNN through the dynamics presented in the "Neural network architecture" section.

### Introducing computation noise in neural networks

RNNs with computation noise feature random noise in the equations that govern its dynamics. We implemented computation noise in the network dynamics by updating the activity of each unit in the network in an imprecise fashion. Let $Z_{t-1}$ be the recurrent activity of the network at time $t-1$ and $\widehat{Z}_t$ the updated state at time $t$ before the nonlinearity $h_Z$. The updated state of each unit in the network is corrupted with independent and identically distributed Gaussian noise. More precisely, we sampled the noisy updated state $\widehat{Z}_{t,k}^{\text{noisy}}$ from a normal distribution of mean equal to the result of the exact (noise-free) update $\widehat{Z}_{t,k}$ and of SD $\sigma$

$$\widehat{Z}_{t,k}^{\text{noisy}} \sim \mathcal{N}\left(\widehat{Z}_{t,k}, \sigma\right) \tag{12}$$

where $\widehat{Z}_{t,k}$ is the activity of unit $k$ ($k \leq K$, where $K = 48$ is the total number of units in the network) such that $\widehat{Z}_t = \left\{\widehat{Z}_{t,1}, \ldots, \widehat{Z}_{t,K}\right\}$. Computation noise at the unit level thus has the same structure in the two tasks. This constant-scaling noise could reflect, at least, in part, task-irrelevant input that effectively corrupts the computation of task-relevant variables. Once the noisy updated state $\widehat{Z}_{t,k}^{\text{noisy}}$ is sampled, the nonlinear activation function $h_Z$ is applied as in exact neural networks.

### Experimental procedures for simulations of the weather prediction task

The training task (task A) is composed of independent trials of $n = 5$ samples of the same cue (sampled uniformly among the eight cues shown in Fig. 2. On each trial, the agent is presented with one of the cues for five samples, after which the agent has to choose between two actions. The agent receives a positive (+1) or negative (−1) reward as a function of the probabilistic association between the presented cue and the chosen action. In task A, optimal behavior consists in choosing the "greedy" action that maximizes the probability of obtaining a positive reward.

Once the network is trained through backpropagation (as described in the "Training procedure" section), we fix its weights and test it in the weather prediction condition (task A*). On each trial,

the agent is now presented with sequences of different cues on each sample, with $n = [2, \ldots, 16]$ samples. In this second task, each cue taken in isolation is associated with the same reward probabilities as in the first task, but reward probabilities can be combined across presented cues to identify the best action associated with the sequence of cues as a whole.

In practice, to determine which cues would be presented on a particular trial, we defined two categories, each associated with one of the two actions being rewarded at the end of the trial. At the beginning of the trial, one of the two categories was randomly selected for subsequent sampling. The distributions of cues associated with each category were defined such that the conditional probability that a cue is sampled matches its reward probability experienced in task A.

### Experimental procedures for simulations of the bandit task

The training task consists of a two-armed bandit game. On each trial of a game, the agent is presented with a slot machine with two levers to choose from. The agent receives a positive (+1) or negative (−1) reward as a function of the reward probability associated with the chosen lever in the current game (0.95 for the best lever and 0.05 for the worst lever), which remains fixed over the course of the game. The most rewarded lever is reset randomly at the beginning of each game, such that the agent needs to learn which lever is most rewarded in every single game.

Once the network is trained through backpropagation (as described in the "Training procedure" section), we fix its weights and test it in several variants of the canonical bandit task described above. In the first variant A* (reversal learning task), we assume games of 50 trials and introduced a reversal in the reward probabilities associated with the two arms in the middle of the game. The false feedback probability in this first variant is set to 0.05, identical to the one in the training task A. In the second variant A*, we introduced a "volatile" condition of 100 trials in which the reward probabilities associated with the two arms switch every 25 trials. Following previous work (*27*), the false feedback is set to 0.2. This volatile is contrasted with a stable condition with no reversal and matching false feedback probability. In the third variant A* (restless bandit task), the reward probabilities associated with the two arms randomly drifts over the course of the 200 trials of each game.

### Ideal Bayesian model in the weather prediction task

In a trial of length $n$, the ideal decision-making observes $n$ cues $s_t$ with $t \in [1, n]$. For each cue $s_t$, the decision-maker computes the log likelihoods $l_{tk} = \log p(s_t \mid C_k)$ that it was generated from either category $k$ (given by Fig. 2A). $k$ here can be 1 or 2, corresponding to the two categories. To determine the preference for a category, the decision-maker calculates the log-likelihood ratio associated with each presented cue

$$r_t = l_{t1} - l_{t2} \tag{13}$$

These log-likelihood ratios are then accumulated over all $n$ cues to form a cumulative log-likelihood ratio

$$z = \sum_{t=1}^{n} r_t \tag{14}$$

The decision-maker then selects the category on the basis of the sign of the cumulative log-likelihood ratio $z$. This method integrates the evidence provided by the cues to make a categorical decision in an optimal way, relying on the different strengths of log-likelihood ratios pointing toward each of the two categories.

### Estimating psychophysical kernels from RNN behavior in the weather prediction task

We performed logistic regressions of RNN behavior to estimate associated psychophysical kernels, both across time (cue position) and across cues (cue identity) in a sequence. For the psychophysical kernel across time, for every sequence length $n$ (number of cues), we performed a logistic regression of the chosen action $a_t$ at trial $t$ as a weighted sum of the evidence provided by each cue $e_{k,t}$ where $k$ is the position of the cue in the sequence. The evidence provided by each cue corresponds to the log ratio between the likelihood of the cue given that $a_t = 1$ and the likelihood of the same cue given that $a_t = 2$ (which is termed log-likelihood ratio). The psychophysical kernel across time corresponds to the estimated weights $w_{k,t}$ assigned to the evidence provided by each cue. We used the logistic function as transfer function for computing the probability of $a_t = 1$ given the weighted sum of the evidence provided by each cue. For the psychophysical kernel across cues, we performed a similar logistic regression of the chosen action $a_t$ at trial $t$ as a weighted sum of the number of times $n_{i,t}$ that each cue $i$ has been presented in the sequence, where $i$ is the identity of the cue ($i$ ranging from 1 to 8). The subjective reliability of cue $i$ corresponds to the estimated weight $w_{i,t}$ assigned to this cue in the logistic regression model, which we plot against the objective reliability of the same cue, which corresponds to the log-likelihood ratio associated with the cue.

### Obtaining the variability scaling with cue uncertainty

With the notations of the "Neural network architecture" section, we denote $Z_T$ the recurrent activity (after the recurrent nonlinearity $h_Z$) at the decision step and $V$ the output matrix. Because we have two choices, A and B, we can write $V$ as the concatenation of two vectors $V_A$ and $V_B$ corresponding to the two possible actions. Upon the presentation of a cue, such as a circle, we calculate the difference in output activation toward actions A and B, expressed as $V_A \cdot Z_T - V_B \cdot Z_T$. Given that half of the cues point toward A and the other half points toward B, we take the absolute value of the difference ensuring that the differences for all cues are positive and comparable. The plot displays these output activations defined as the average of $|V_A \cdot Z_T - V_B \cdot Z_T|$ alongside the output variability, defined as the SD of $|V_A \cdot Z_T - V_B \cdot Z_T|$.

### Estimating the learning rate from RNN behavior in the volatility task

We fitted the standard Rescorla-Wagner (RW) model in the volatile and stable condition independently. This model tracks expected rewards ($Q$ values) and has with two free parameters: a learning rate $\alpha$ and a softmax parameter $\beta$. Given an action $a_t$ and reward $r_t \in \{0,1\}$, the $Q$ value associated to the action $a_t$ is updated such that

$$Q_{t+1} = Q_t + \alpha \cdot \left(r_t - Q_t\right) \tag{15}$$

where $\alpha$ is the learning rate used to update action values based on the PE between obtained reward $r_{t-1}$ and expected reward $Q_{t-1}$ on the previous trial. The value associated with the unchosen arm is also updated by assuming a fictive counterfactual reward (when $r_t = 0$, then counterfactual reward is 1 and conversely) as in (*27*). The learning rate $\alpha$ controls the rate of integration: The larger the $\alpha$ value, the

more weight the rule gives to recent observations. As in existing theories, we modeled the choice process using a stochastic softmax action selection policy, controlled by an inverse temperature β

$$a_t \sim \mathrm{B}\left\{\frac{1}{1+\exp\left[-\beta\left(Q_{t,\mathrm{A}}-Q_{t,\mathrm{B}}\right)\right]}\right\} \quad (16)$$

where B(. ) denotes the Bernoulli distribution and $Q_{t,\mathrm{A}}$ and $Q_{t,\mathrm{B}}$ correspond to the values associated with actions A and B, respectively. This stochastic action selection policy reduces to a purely greedy (value-maximizing) "argmax" policy when $\beta \to \infty$.

### Obtaining the computation noise as a function of the quantity of update

Using the notation of the "Neural network architecture" and "Introducing computation noise in neural networks" sections, we used the recurrent activity before and after noise addition, denoted as $\widehat{Z}_t$ and $\widehat{Z}_t^{\mathrm{noisy}}$, respectively. These values are passed through the recurrence's nonlinearity $h_Z$, represented as $Z_t = h_Z\left(\widehat{Z}_t\right)$ and $Z_t^{\mathrm{noisy}} = h_z\left(\widehat{Z}_t^{\mathrm{noisy}}\right)$. Let $V$ be the output matrix. It can be written as the concatenation of two vector $V_\mathrm{A}$ and $V_\mathrm{B}$, respectively, associated with actions A and B. Subsequently, $Z_t$ and $Z_t^{\mathrm{noisy}}$ are projected onto the decision axis as $\Delta V \cdot Z_t = (V_\mathrm{A} \cdot Z_t - V_\mathrm{B} \cdot Z_t)$ and $\Delta V \cdot Z_t^{\mathrm{noisy}} = \left(V_\mathrm{A} \cdot Z_t^{\mathrm{noisy}} - V_\mathrm{B} \cdot Z_t^{\mathrm{noisy}}\right)$. The update magnitude projected onto the decision axis is defined as $\left|\Delta V \cdot Z_t - \Delta V \cdot Z_{t-1}^{\mathrm{noisy}}\right|$. Regarding the computation noise projected on the decision axis, it is defined as $\left|\Delta V \cdot Z_t^{\mathrm{noisy}} - \Delta V \cdot Z_t\right|$. To visualize these quantities, we categorized all quantities of updates into 10 bins and computed the median computation noise for each bin and agent. Using the median helps mitigate the influence of extreme values, providing a clearer indication of the typical value of computation noise. The resulting plot displays the average along with the SD of these medians across all $n = 50$ RNNs.

### Fitting noisy RW models to RNN and human behavior

To characterize the amount of computation noise, we used the RW rule where the updating of the expected reward $Q_{t-1}$ associated with the chosen action $a_{t-1}$ is corrupted by additive random noise $\varepsilon_t$

$$Q_t = Q_{t-1} + \alpha\left(r_{t-1} - Q_{t-1}\right) + \varepsilon_t \quad (17)$$

where α is the learning rate used to update action values based on the PE between obtained reward $r_{t-1}$ and expected reward $Q_{t-1}$ on the previous trial and $\varepsilon_t$ models the stochastic deviation from the exact rule. Again, as explained in the "Estimating the learning rate from RNN behavior in the volatility task" section, the value associated with the unchosen action was also updated assuming counterfactual fictive rewards and corrupted by additive random noise. This "unchosen" additive random noise is independent but is defined and sampled in a similar way as the random noise associated with the chosen action (*7*).

When fitting RW models to the RNNs simulated behavior and the human behavior, we considered three types of models. The first assumes no noise, meaning that $\varepsilon_t = 0, \forall\, t$. In this case, we obtain exactly the model used to estimate the learning rate. The second assumes white noise, meaning that $\varepsilon_t$ is sampled from a normal distribution with zero mean and SD σ (treated as a free parameter). Last, the third assumes Weber noise, and $\varepsilon_t$ is drawn from a normal distribution with zero mean and SD $\sigma_t$ equal to a constant fraction ζ (treated as a free parameter) of the magnitude of the PE: $\sigma_t = \zeta\,|r_{t-1} - Q_{t-1}|$.

### Statistical tests

Throughout the manuscript, we used various statistical tests to determine the significance of differences between groups of RNNs. Specifically, we used the *t* test for comparing the means of two groups. For comparing differences across multiple conditions while accounting for within-agent correlations, we used repeated-measures ANOVA. To assess correlations, we used Pearson's R correlation.

## Supplementary Materials

**This PDF file includes:**

Figs. S1 to S8

## REFERENCES AND NOTES


1. A. A. Faisal, L. P. J. Selen, D. M. Wolpert, Noise in the nervous system. *Nat. Rev. Neurosci.* **9**, 292–303 (2008).
2. V. Wyart, E. Koechlin, Choice variability and suboptimality in uncertain environments. *Curr. Opin. Behav. Sci.* **11**, 109–115 (2016).
3. C. Findling, V. Wyart, Computation noise in human learning and decision-making: Origin, impact, function. *Curr. Opin. Behav. Sci.* **38**, 124–132 (2021).
4. R. Moreno-Bote, J. Rinzel, N. Rubin, Noise-induced alternations in an attractor network model of perceptual bistability. *J. Neurophysiol.* **98**, 1125–1139 (2007).
5. B. van Vugt, B. Dagnino, D. Vartak, H. Safaai, S. Panzeri, S. Dehaene, P. R. Roelfsema, The threshold for conscious report: Signal loss and response bias in visual and frontal cortex. *Science* **360**, 537–542 (2018).
6. J. Drugowitsch, V. Wyart, A.-D. Devauchelle, E. Koechlin, Computational precision of mental inference as critical source of human choice suboptimality. *Neuron* **92**, 1398–1411 (2016).
7. C. Findling, V. Skvortsova, R. Dromnelle, S. Palminteri, V. Wyart, Computational noise in reward-guided learning drives behavioral variability in volatile environments. *Nat. Neurosci.* **22**, 2066–2077 (2019).
8. C. Findling, N. Chopin, E. Koechlin, Imprecise neural computations as a source of adaptive behaviour in volatile environments. *Nat. Hum. Behav.* **5**, 99–112 (2021).
9. H. B. Barlow, "Possible principles underlying the transformations of sensory messages" in *Sensory Communication*, W. A. Rosenblith, Ed. (MIT Press, 1961), pp. 217–234.
10. B. B. Averbeck, P. E. Latham, A. Pouget, Neural correlations, population coding and computation. *Nat. Rev. Neurosci.* **7**, 358–366 (2006).
11. X.-X. Wei, A. A. Stocker, A Bayesian observer model constrained by efficient coding can explain "anti-Bayesian" percepts. *Nat. Neurosci.* **18**, 1509–1517 (2015).
12. R. Polanía, M. Woodford, C. C. Ruff, Efficient coding of subjective value. *Nat. Neurosci.* **22**, 134–142 (2019).
13. A. Tversky, D. Kahneman, Judgment under uncertainty: Heuristics and biases. *Science* **185**, 1124–1131 (1974).
14. G. Gigerenzer, R. Selten, *Bounded Rationality: The Adaptive Toolbox* (MIT Press, 2002).
15. D. C. Knill, A. Pouget, The Bayesian brain: The role of uncertainty in neural coding and computation. *Trends Neurosci.* **27**, 712–719 (2004).
16. M. Oaksford, N. Chater, *Bayesian Rationality: The Probabilistic Approach to Human Reasoning* (Oxford Univ. Press, 2007).
17. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**, 1929–1958 (2014).
18. L. Gammaitoni, P. Hänggi, P. Jung, F. Marchesoni, Stochastic resonance. *Rev. Mod. Phys.* **70**, 223–287 (1998).
19. B. Lindner, J. García-Ojalvo, A. Neiman, L. Schimansky-Geier, Effects of noise in excitable systems. *Phys. Rep.* **392**, 321–424 (2004).
20. B. A. Richards, T. P. Lillicrap, P. Beaudoin, Y. Bengio, R. Bogacz, A. Christensen, C. Clopath, R. P. Costa, A. de Berker, S. Ganguli, C. J. Gillon, D. Hafner, A. Kepecs, N. Kriegeskorte, P. Latham, G. W. Lindsay, K. D. Miller, R. Naud, C. C. Pack, P. Poirazi, P. Roelfsema, J. Sacramento, A. Saxe, B. Scellier, A. C. Schapiro, W. Senn, G. Wayne, D. Yamins, F. Zenke, J. Zylberberg, D. Therien, K. P. Kording, A deep learning framework for neuroscience. *Nat. Neurosci.* **22**, 1761–1770 (2019).
21. A. Saxe, S. Nelli, C. Summerfield, If deep learning is the answer, what is the question? *Nat. Rev. Neurosci.* **22**, 55–67 (2021).
22. V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, K. Kavukcuoglu, Asynchronous Methods for Deep Reinforcement Learning, in *Proceedings of The 33rd International Conference on Machine Learning*, New York, USA, 20 to 22 June 2016 (PMLR, 2016), vol. 48, pp. 1928–1937.

23. J. X. Wang, Z. Kurth-Nelson, D. Kumaran, D. Tirumala, H. Soyer, J. Z. Leibo, D. Hassabis, M. Botvinick, Prefrontal cortex as a meta-reinforcementlearning system. *Nat. Neurosci.* **21**, 860–868 (2018).
24. B. J. Knowlton, J. A. Mangels, L. R. Squire, A neostriatal habit learning system in humans. *Science* **273**, 1399–1402 (1996).
25. R. A. Poldrack, J. Clark, E. J. Paré-Blagoev, D. Shohamy, J. Creso Moyano, C. Myers, M. A. Gluck, Interactive memory systems in the human brain. *Nature* **414**, 546–550 (2001).
26. T. Yang, M. N. Shadlen, Probabilistic reasoning by neurons. *Nature* **447**, 1075–1080 (2007).
27. T. E. J. Behrens, M. W. Woolrich, M. E. Walton, M. F. S. Rushworth, Learning the value of information in an uncertain world. *Nat. Neurosci.* **10**, 1214–1221 (2007).
28. J. Hernández-Orallo, *The Measure of All Minds: Evaluating Natural and Artificial Intelligence* (Cambridge Univ. Press, 2017).
29. F. Chollet, On the measure of intelligence. arXiv:1911.01547 (2019).
30. B. M. Lake, T. D. Ullman, J. B. Tenenbaum, S. J. Gershman, Building machines that learn and think like people. *Behav. Brain Sci.* **40**, e253 (2017).
31. A. Pouget, J. M. Beck, W. J. Ma, P. E. Latham, Probabilistic brains: Knowns and unknowns. *Nat. Neurosci.* **16**, 1170–1178 (2013).
32. S. Iglesias, C. Mathys, K. H. Brodersen, L. Kasper, M. Piccirelli, H. E. M. den Ouden, K. E. Stephan, Hierarchical prediction errors in midbrain and basal forebrain during sensory learning. *Neuron* **80**, 519–530 (2013).
33. M. M. Botvinick, Y. Niv, A. G. Barto, Hierarchically organized behavior and its neural foundations: A reinforcement learning perspective. *Cognition* **113**, 262–280 (2009).
34. M. K. Eckstein, A. G. E. Collins, Computational evidence for hierarchically structured reinforcement learning in humans. *Proc. Natl. Acad. Sci. U.S.A.* **117**, 29381–29389 (2020).
35. M. Heilbron, F. Meyniel, Confidence resets reveal hierarchical adaptive learning in humans. *PLOS Comput. Biol.* **15**, e1006972 (2019).
36. C. Foucault, F. Meyniel, Gated recurrence enables simple and accurate sequence prediction in stochastic, changing, and structured environments. *eLife* **10**, e71801 (2021).
37. V. Mante, D. Sussillo, K. V. Shenoy, W. T. Newsome, Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature* **503**, 78–84 (2013).
38. D. Peixoto, J. R. Verhein, R. Kiani, J. C. Kao, P. Nuyujukian, C. Chandrasekaran, J. Brown, S. Fong, S. I. Ryu, K. V. Shenoy, W. T. Newsome, Decoding and perturbing decision states in real time. *Nature* **591**, 604–609 (2021).
39. D. P. Kingma, M. Welling, Auto-encoding variational Bayes. arXiv:1312.6114 (2022).
40. M. Fortunato, M. G. Azar, B. Piot, J. Menick, I. Osband, A. Graves, V. Mnih, R. Munos, D. Hassabis, O. Pietquin, C. Blundell, S. Legg, Noisy networks for exploration. arXiv:1706.10295 (2017).
41. M. Plappert, R. Houthooft, P. Dhariwal, S. Sidor, R. Y. Chen, X. Chen, T. Asfour, P. Abbeel, M. Andrychowicz, Parameter space noise for exploration. arXiv:1706.01905 (2018).
42. W. J. Ma, V. Navalpakkam, J. M. Beck, R. van den Berg, A. Pouget, Behavior and neural basis of near-optimal visual search. *Nat. Neurosci.* **14**, 783–790 (2011).
43. A. M. Zador, A critique of pure learning and what artificial neural networks can learn from animal brains. *Nat. Commun.* **10**, 3770 (2019).
44. U. Hasson, S. A. Nastase, A. Goldstein, Direct fit to nature: An evolutionary perspective on biological and artificial neural networks. *Neuron* **105**, 416–434 (2020).
45. M. Matsumoto, K. Matsumoto, H. Abe, K. Tanaka, Medial prefrontal cell activity signaling prediction errors of action values. *Nat. Neurosci.* **10**, 647–656 (2007).
46. C. Padoa-Schioppa, J. A. Assad, The representation of economic value in the orbitofrontal cortex is invariant for changes of menu. *Nat. Neurosci.* **11**, 95–102 (2008).
47. M. F. S. Rushworth, M. P. Noonan, E. D. Boorman, M. E. Walton, T. E. Behrens, Frontal cortex and reward-guided learning and decision-making. *Neuron* **70**, 1054–1069 (2011).
48. R. Echeveste, L. Aitchison, G. Hennequin, M. Lengyel, Cortical-like dynamics in recurrent circuits optimized for sampling-based probabilistic inference. *Nat. Neurosci.* **23**, 1138–1149 (2020).
49. J. Dapello, T. Marques, M. Schrimpf, F. Geiger, D. Cox, J. J. DiCarlo, "Simulating a primary visual cortex at the front of CNNs improves robustness to image perturbations" in *Advances in Neural Information Processing Systems* (Curran Associates Inc., 2020), vol. 33, pp. 13073–13087.
50. T. Hastie, R. Tibshirani, M. Wainwright, *Statistical Learning with Sparsity: The Lasso and Generalizations* (Chapman and Hall/CRC, 2019).
51. M. C. M. Faraut, E. Procyk, C. R. E. Wilson, Learning to learn about uncertain feedback. *Learn. Mem.* **23**, 90–98 (2016).
52. M. Lopes, T. Lang, M. Toussaint, P. Oudeyer, "Exploration in model-based reinforcement learning by empirically estimating learning progress" in *Advances in Neural Information Processing Systems (NeurIPS)* (2012), pp. 206–214.
53. M. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, R. Munos, "Unifying count-based exploration and intrinsic motivation" in *Advances in Neural Information Processing Systems (NeurIPS)* (2016), pp. 1471–1479.
54. D. Pathak, P. Agrawal, A. A. Efros, T. Darrell, "Curiosity-driven exploration by self-supervised prediction" *in International Conference on Machine Learning (ICML)* (2017), pp. 2778–2787.
55. A. Dezfouli, H. Ashtiani, O. Ghattas, R. Nock, P. Dayan, C. S. Ong, "Disentangled behavioural representations" in *Advances in Neural Information Processing Systems* (Curran Associates Inc., 2019), vol. 32, pp. 2254–2263.
56. K. J. Miller, M. Eckstein, M. M. Botvinick, Z. Kurth-Nelson, Cognitive Model Discovery via Disentangled RNNs. bioRxiv 546250 [Preprint] (2023). https://doi.org/10.1101/2023.06.23.546250.
57. J. K. Lee, M. Rouault, V. Wyart, Adaptive tuning of human learning and choice variability to unexpected uncertainty. bioRxiv 520751 [Preprint] (2022). https://doi.org/10.1101/2022.12.16.520751.



**Acknowledgments**

**Funding:** This work was supported by a starting grant from the European Research Council (ERC-StG759341) awarded to V.W. and by a department-wide grant from the Agence Nationale de la Recherche (ANR-17-EURE-0017, EUR FrontCog). **Author contributions:** C.F.: conceptualization, methodology, software, validation, formal analysis, investigation, resources, data curation, writing—original draft, writing—review and editing, visualization, and supervision. V.W.: conceptualization, methodology, software, validation, formal analysis, investigation, resources, data curation, writing—original draft, writing—review and editing, visualization, supervision, project administration, and funding acquisition. **Competing interests:** The authors declare that they have no competing interests. **Data and materials availability:** All data needed to evaluate the conclusions in the paper are present in the paper and/or the Supplementary Materials. The code used to train and simulate noisy RNNs is available on a public online repository at https://zenodo.org/doi/10.5281/zenodo.13769737 (and on GitHub at https://github.com/csmfindling/cognitive_resilience). The code used to fit noisy RW models to the behavior of human participants and noisy RNNs in the bandit (reversal learning) task is also available on a public online repository at https://zenodo.org/doi/10.5281/zenodo.13769718 (and on GitHub at https://github.com/csmfindling/learning_variability). The human dataset obtained in a restless two-armed bandit task analyzed in this study is also available on a public online repository at https://doi.org/10.5281/zenodo.12532395.

Submitted 16 October 2023
Accepted 24 September 2024
Published 30 October 2024
10.1126/sciadv.adl3931