PNAS
www.pnas.org

Supplementary information for:

Learning probabilistic neural representations with randomly connected circuits

Ori Maoz, Gašper Tkačik, Mohamad Saleh Esteki, Roozbeh Kiani, Elad Schneidman

Corresponding authors;

- Elad Schneidman: elad.schneidman@weizmann.ac.il
- Roozbeh Kiani: roozbeh@nyu.edu

This PDF file includes:

- Supplementary text
- Figures S1 to S9
- Algorithms S1 to S2
- SI references

www.pnas.org/cgi/doi/10.1073/pnas.1912804117

Self-normalization in RP models

As explained in main text, the membrane potential of the readout neuron $y(\vec{x})$ (Eq. 1) reflects the surprise of the input up to an additive factor, which stems from the normalization term of the model (or partition function). We stress that the basic function we propose for a neural circuit - determining which input patterns are surprising - requires knowing only the relative probabilities of patterns, which are unaffected by normalization. The same applies for implementing a generative model (though other architectures, e.g. Restricted Boltzmann Machines, might be more suitable for that purpose). In many classification problems, one needs to compare the likelihood values of alternatives, which means only the ratio of probabilities matter and the normalization term cancels out. This would imply that the membrane voltage of the readout neurons would be sufficient.

In other cases, such as when comparing the responses of two different neurons, the responses of these neurons must be normalized in order to be comparable. In these situations, the readout neuron can estimate the normalized value of surprise if we consider an additive term to the membrane voltage that is learned through experience. For example, if the neural code is sparse, the neuron can learn this additive term by taking advantage of the fact that for the all-zero input pattern, $\vec{0}$, $p(\vec{0}) = \frac{1}{Z}$ (see [1]) and so the additive factor can be set according to how frequently the neuron receives no spiking input. An alternative would be to consider the spiking of the readout neuron, which would reflect inputs with high surprise, determined by the spiking threshold of the cell. In terms of the spiking of the readout neuron, changing its threshold would be equivalent to an additive term to the membrane potential. As neurons employ homeostatic mechanisms to adjust their activity rates to certain ranges [2], this could be a self-normalizing mechanism for estimating the surprise.

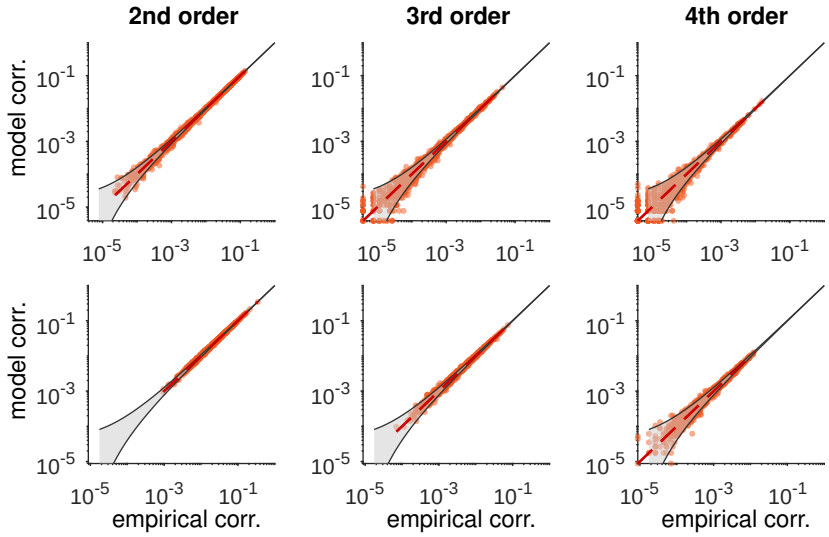

Figure S1: Correlations $\langle x_i x_j x_k \rangle_{\hat{p}}$ in test data vs. model prediction for RP models trained over population activity patterns of 178 cells from the visual cortex (top) and 169 cells from prefrontal cortex (bottom).

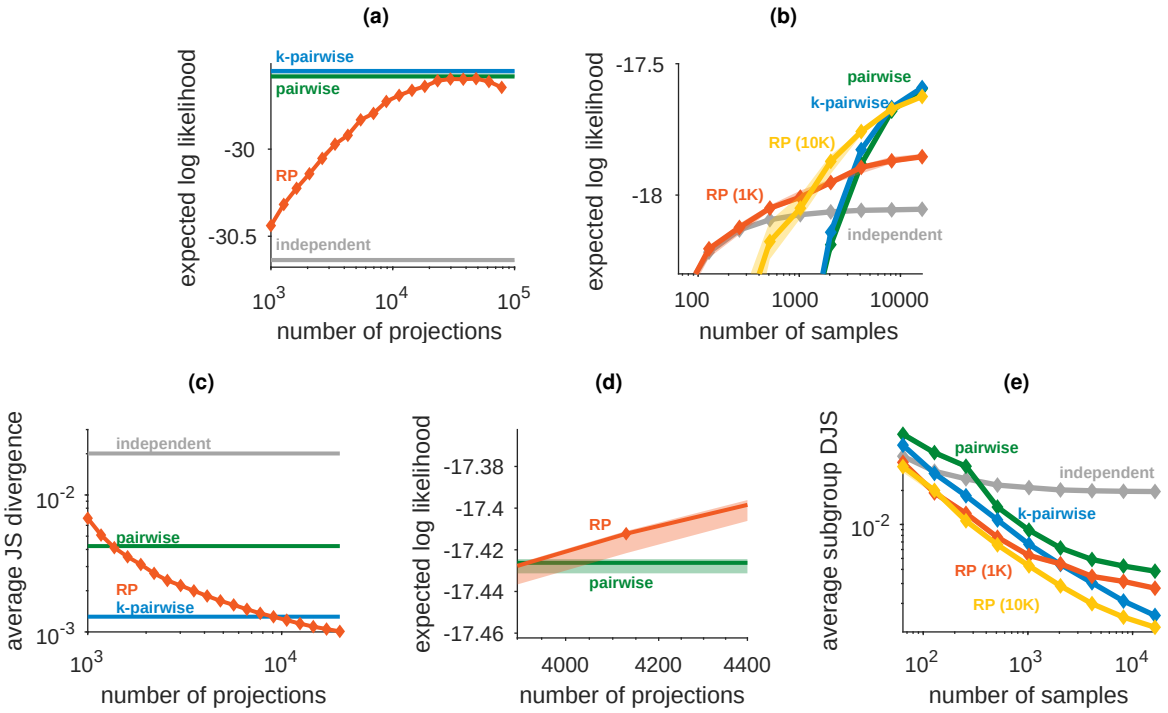

Figure S2: (**a**) Expected likelihood of probabilistic models trained from 100,000 population activity patterns from 169 neurons in the prefrontal cortex as a function of the number of projections in the RP model. The performance of the RP models begin to deteriorate as the number of projections approaches the number of training samples. Error bars denote standard deviation across random choices of projections and train/test divisions. **(b)** Expected likelihood of RP trained on population activity patterns of 100 neurons from the monkey prefrontal cortex as a function of the number of samples in the training data, for RP models using 1,000 random projections (*RP 1K*), 10,000 random projections (*RP 10K*), pairwise, k-pairwise and independent models. **(c)** Average Jensen-Shannon divergence between model and test data for randomly selected subgroups of 10 neurons in a population of 178 neurons from the visual cortex - same data and models as Fig. 3a. **(d)** Zoomed-in version of Fig. 3a shows the low variability of RP models for different instantiations of the random projections and random choices of train/test data. **(e)** Performance of probabilistic models trained over population activities of 100 neurons from the primate visual cortex, as average Jensen-Shannon divergence between model and test data for randomly selected subgroups of 10 neurons. - same data and models as Fig. 3c.

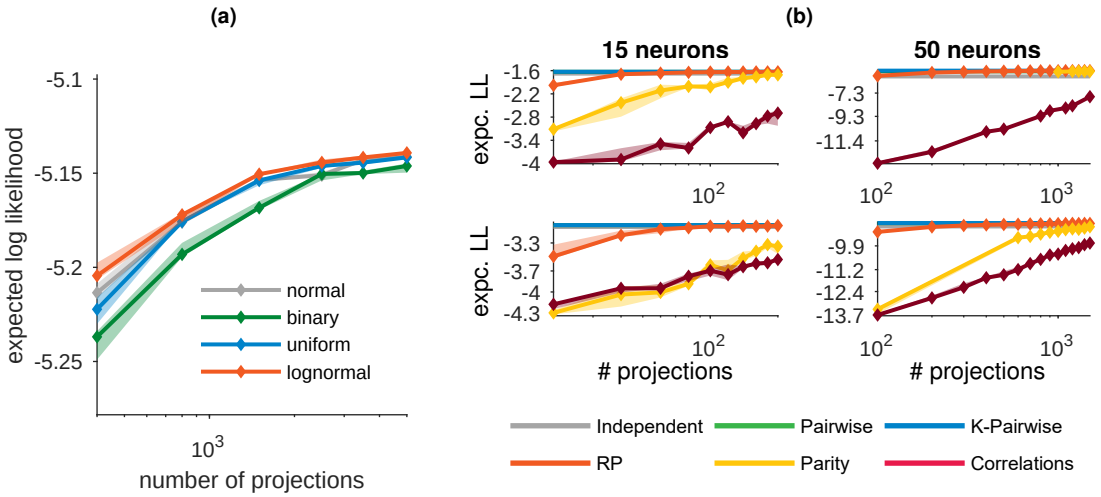

Figure S3: (a) RP model performance for different choices of the distributions of the random projection weights, trained over joint activity patterns of 50 neurons from the primate visual cortex. **(b)** Performance of the RP model compared to models based on high-order correlations and high-order parities of the data, for data from the primate visual cortex (top) or prefrontal cortex (bottom).

Other choices of random projection statistics and random function families

RP models were largely unaffected by how the random projection elements were chosen. Fig. S3a shows the cross-validated performance, over experimental data, for models where the synaptic weights $a_{i,j}$ were drawn from: (1) Normal distribution ($\mu = 1, \sigma = 1$), (2) Log-normal distribution ($\mu = 0, \sigma = 1$), (3) Uniform distribution in the range $[-0.2....0.8]$ and (4) Binary distribution ($p(1) = 0.8$, $p(-1) = 0.2$). We found that the different models performed similarly for the data at hand, with the normal and log-normal variants slightly outperforming the others.

We also examined two alternative choices of random function families for a probabilistic model: randomly selected high-order correlations and randomly selected high-order parities. Probabilistic models of randomly selected high-order correlations are maximum entropy distributions constrained over a random selection of high-order correlations, $\langle \prod_{j \in C_i} x_j \rangle$, where $C_1 ... C_k$ are randomly chosen groups of neurons in the population. This gives a probabilistic model of the form:

$$\hat{p}(x) = \frac{1}{Z} \exp(\sum_{i=1}^{k} \lambda_i \prod_{j \in C_i} x_j) \tag{S1}$$

Probabilistic models of randomly selected high-order parities are maximum entropy distributions constrained over the mean parities (XOR) of the activities of randomly selected groups of neurons, $\langle \bigoplus_{j \in C_i} x_j \rangle$. This gives a probabilistic model of the form:

$$\hat{p}(x) = \frac{1}{Z} \exp(\sum_{i=1}^{k} \lambda_i \bigoplus_{j \in C_i} x_j) \tag{S2}$$

We found that these two models under-performed in comparison to the RP model (Fig. S3b) when

trained over our experimental recordings. The particularly poor performance of the correlation-based model can be attributed to the fact that measuring high-order correlations is unreliable when the activity patterns are sparse, as is typically the case in spiking neurons.

RP models capture high-order interactions

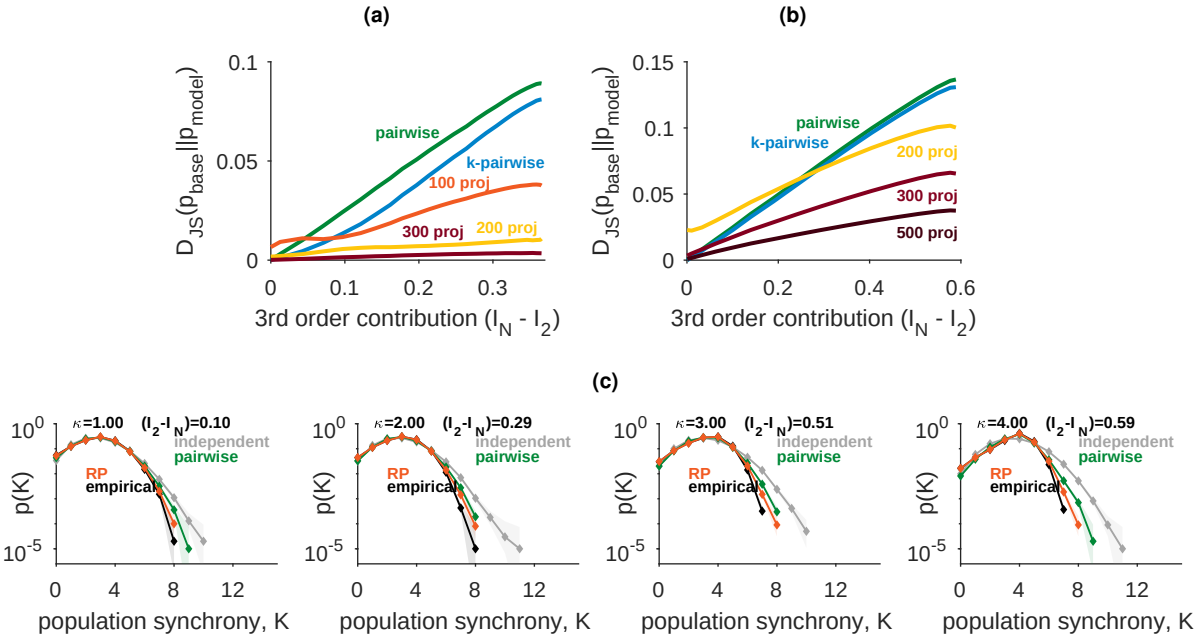

Figure S4: Performance of maximum entropy models as a function of the strength of high-order interactions in small artificial population activity distributions of **(a)** ten and **(b)** fifteen neurons. Pairwise ME models and pairwise with synchrony constraints perfectly captured data generated from pairwise distributions and quickly degraded as stronger third-order interactions were introduced. Random models provided relatively good results even when the third-order interactions were strong. **(c)** Population synchrony taken from (b) where κ ranges from 1 (left) to 4 (right), evaluated using 100,000 artificially-generated samples of groups of 15 neurons.

We tested whether RP models can successfully learn data simulated from artificial probability distribution with known high-order interactions by creating a family of parameterized Boltzmann distributions of the form:

$$p(x) = \frac{1}{Z} \exp\left[-\sum_{i=1}^{n} \alpha_i x_i - \sum_{(i,j)\in B} \beta_{ij} x_i x_j - \kappa \cdot \sum_{(i,j,k)\in C} \gamma_{ijk} x_i x_j x_k \right] \tag{S3}$$

Where B, C denote randomly selected pairs and triplets of neurons respectively, and the values $\alpha_i, \beta_{ij}, \gamma_{ijk}$ were selected randomly from normal distributions. This results in a family of distributions with pairwise interactions when $\kappa = 0$ and increasingly strong third-order interactions for larger values of κ. These distributions were used to generate simulated data, from which RP models were trained. Fig. S4 shows the Jensen-Shannon divergence between the trained models and the original distribution as a function of $I_N - I_2$, the amount of information in the distribution which cannot be described by pairwise interactions. Maximum entropy models based on pairwise interactions trivially managed to learn when $\kappa = 0$ but made increasingly large errors as stronger third-order interactions where introduced (larger values of κ). RP models were able to capture the high-order interactions more successfully (Fig. S4). The biases in population synchrony exhibited by RP models for this

synthetic data were qualitatively similar to those of pairwise models, though of lesser extent (Fig. S4c).

RP models trained over population activity patterns of actual neural recordings were able to closely capture high-order correlations in the code. Fig. S1 shows 2nd, 3rd and 4th order correlations in the data and as predicted by RP models trained over a separate training set for neural recordings from the primate visual cortex and PFC.

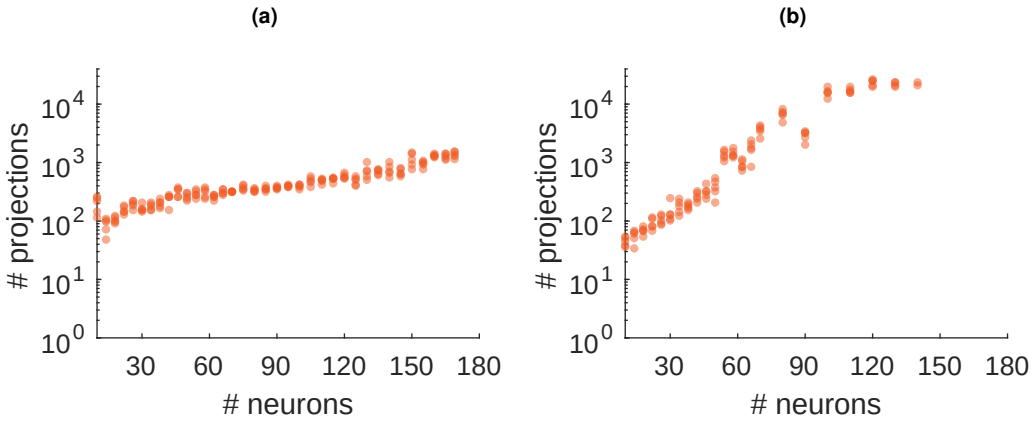

Figure S5: Number of projections required to maintain a consistent model performance. The minimum number of projections required to obtain $DJS_K(model, data) \leq 3 \cdot DJS_K(data, data)$, For increasingly large neuron groups of the **(a)** prefrontal cortex and **(b)** visual cortex, when training RP models with 10,000 activity patterns.

Scaling of the random projections with the input size

We examined how the number of random projections used by the RP model scales with the dimensionality of the input, by seeking the numbers of projections required to maintain a similar-quality fit across different input sizes. We trained RP models over inputs of increasing sizes $N = 10, ..., 70$ and evaluated the probability of the simultaneous activation of K neurons (population synchrony) for $K = 0, 1, ..., N$ under each model. We then evaluated how well each model fits the experimental data by computing $DJS_K(model, data)$: the Jensen-Shannon divergence between the population synchrony of the model and that of the test data. We also computed the median value of $DJS_K(data, data)$, the Jensen-Shannon divergence between random selections from the experimental data, which also corresponds to the DJS_K that would be obtained for a model that perfectly fits the data. Finally, for each N, we searched for the minimum number of projections required to obtain $DJS_K(model, data) < r \cdot DJS_K(data, data)$, where $r > 1$ is some constant.

Fig. S5 shows the number of projections required to maintain a consistant fit (with $r = 3$) across different input sizes, when trained over joint population activity patterns from the visual cortex and the prefrontal cortex. We observed that the number of projections scales differently between the two datasets: while in both cases the increase in the number of projections appears roughly exponential, it rises considerably slower in the prefrontal data than in the visual cortex data. We stress that these results should be taken with a grain of salt, as it is difficult to control all aspects of model convergence across different values of N, and there many other possible choices for defining a "consistent fit". In particular, we would caution against extrapolating from these results to large values of N.

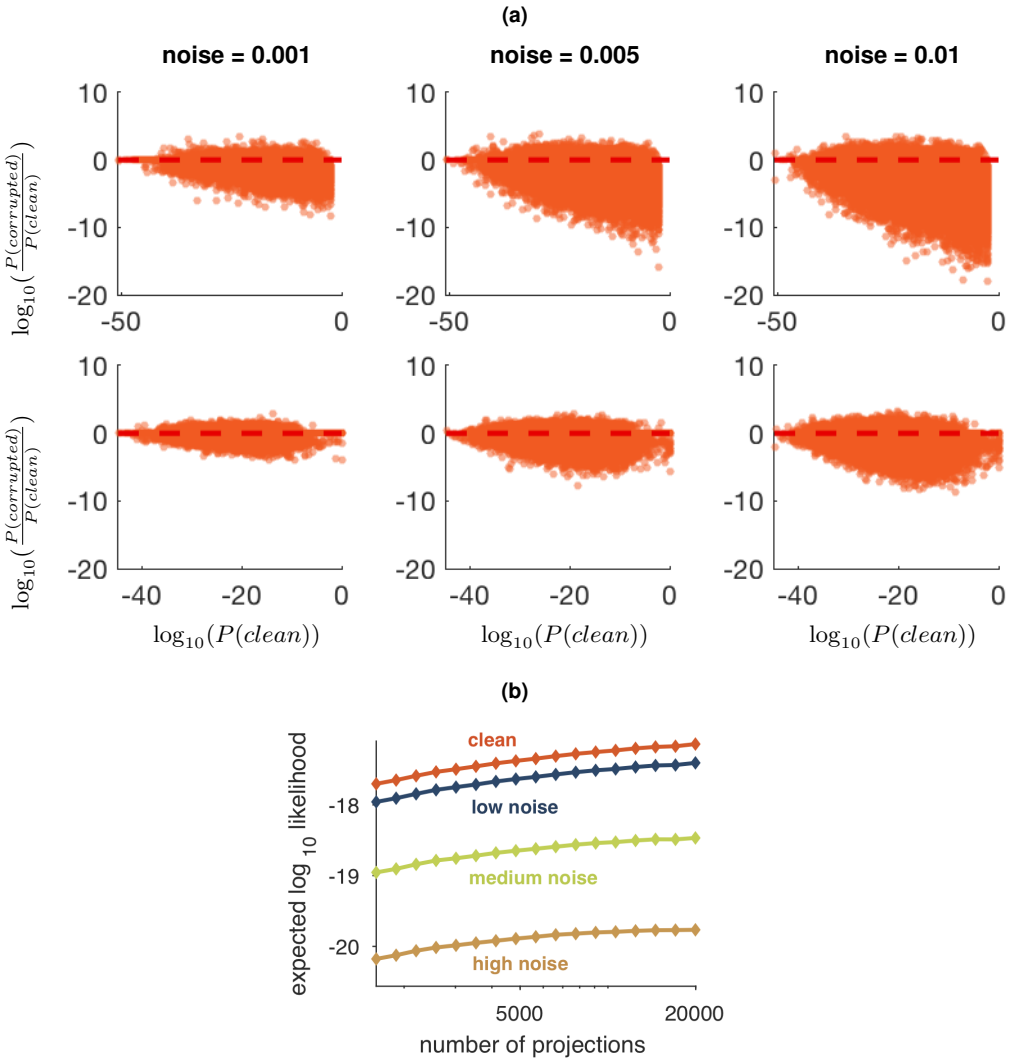

Figure S6: (a) Relative change in the likelihood of test data when it is corrupted by noise. We evaluated pre-trained RP models over 178 neurons from the visual cortex (top) and 169 neurons from the prefrontal cortex (bottom) on test data in which each bit has been randomly and independently flipped with probability 0.1% (left), 0.5% (middle) and 1% (right). Dotted red line denotes no change in likelihood following the bit flip. **(b)** Likelihood of test data over RP models trained over 178 neurons from the visual cortex, with varying amounts random projections, using uncorrupted test data ("clean"), and test data in which each bit has been randomly flipped with probability of 0.001 ("low noise"), 0.005 ("medium noise") and 0.01 ("high noise). All models were trained over uncorrupted training data.

Effects on noise on the likelihood of test data

We examined the effects that noise has on the likelihood of our neural data over RP models. Fig. S6a shows how the likelihoods of individual activity patterns of test data change as they are corrupted by applying random bit-flipping noise to each input dimension independently (some patterns remain unchanged, as do their likelihoods, due to the nature of independent noise). The likelihood

of individual patterns typically decreases as they are corrupted by noise, which is expected as there are vastly more uncommon patterns than common ones. In our datasets, the effect was more pronounced in data recorded from the visual cortex than of data from the prefrontal cortex. This could be explained by the stronger sparsity of the visual cortical activity patterns, that would cause most random bit flips to be $0 \rightarrow 1$ into a less-sparse pattern. The mean likelihood of the test data scaled smoothly with the magnitude of noise (Fig. S6b). We note that the difference between the likelihoods of a clean and corrupted activity pattern is a key element in the learning rule presented in this paper.

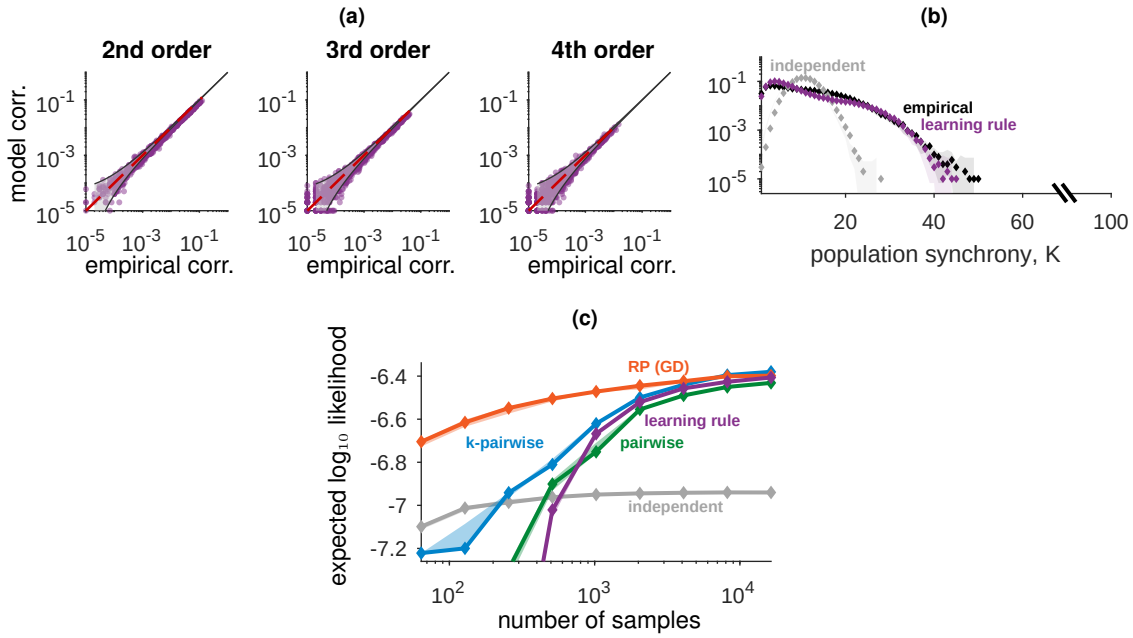

Figure S7: Probabilistic models trained with the biologically plausible learning rule. **(a)** Correlations in test data vs. model prediction for RP models trained with the learning rule over population activities of 100 neurons from the primate visual cortex. **(b)** Population synchrony of RP models trained with the learning rule on groups of 100 neurons, compared to empirical data and reference models. **(c)** Expected likelihood of probabilistic models trained from population activity patterns of 50 neurons from the primate visual cortex as a function of the number of training samples available. In purple: RP model trained with the online learning rule.

Biological interpretation of the ‘echo’ patterns

As explained in the main text, the synaptic learning rule for the RP model (Eq. 1) compares the circuit’s response to the input with its response to a noisy ‘echo’ of the input. Such echo patterns can be generated by biophysical noise either in the neurons or synapses [3]. Updating the synaptic weights to the output neuron would require a back-propagating signal from the cell body [4] and a mechanism that would allow comparing between the synapses’ current and recent activities [5], short-term memory within cells [6, 7], or more complicated local synaptic computations [8]. We note that neural activity during sleep has been characterized with a replay of neural activity statistics alongside highly regular oscillations of neural population activity [9, 10], which could generate replayed inputs with periodically added noisy echos.

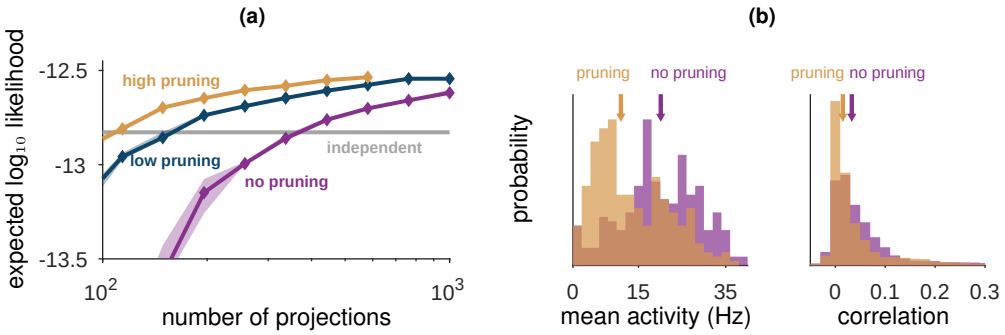

Figure S8: (a) Expected log likelihood of RP models trained with the local learning rule on population activity patterns of 70 neurons from the monkey prefrontal cortex (similar to the visual cortex results in Fig. 5a), while periodically pruning weak synapses and replacing them with new randomly chosen projections. Curves denote the performance of models trained with different total average number of replacements per synapse (low: 2, high: 8). **(b)** Average firing rates (left) and Pearson correlations (right) of intermediate units h_i in models trained with the learning rule with (orange) or without (purple) pruning and replacement; arrows denote median values.

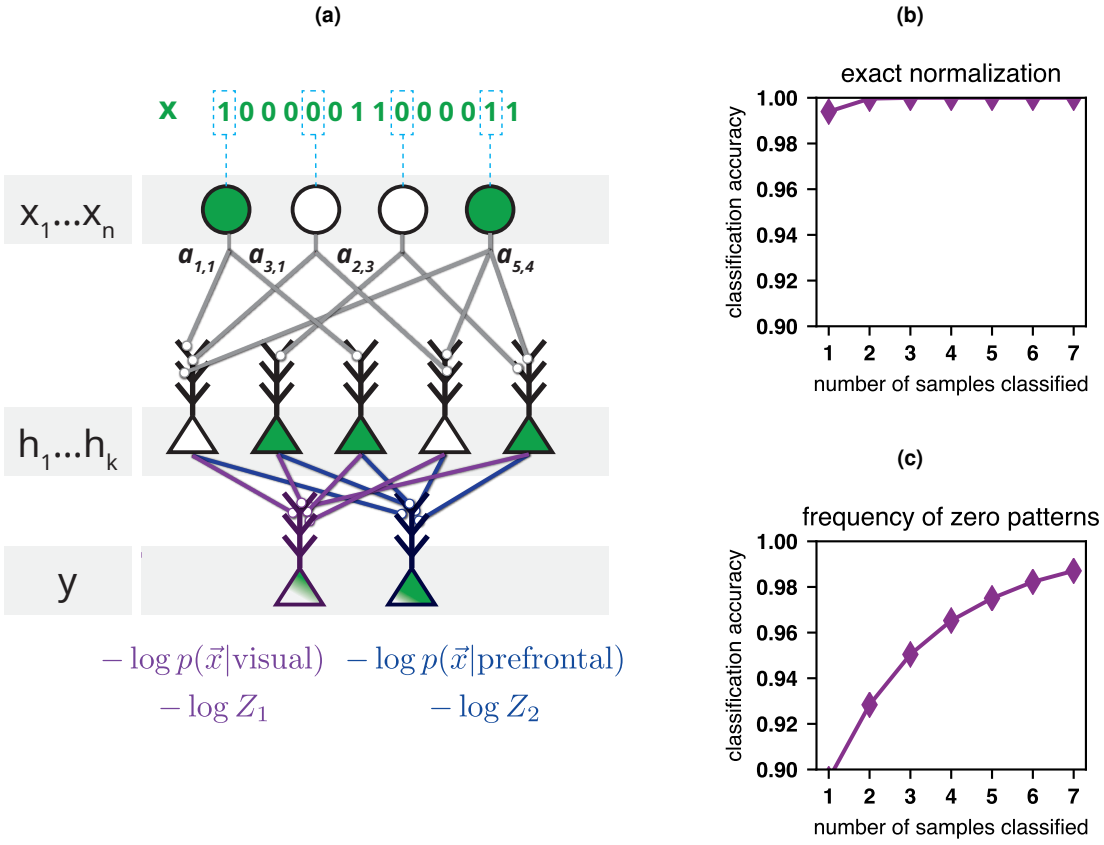

Figure S9: (**a)** An RP circuit with two output neurons that classifies its inputs as coming from either the prefrontal cortex or visual cortex, after applying the learning rule for 10 epochs using 100,000 samples from each class. By selectively applying the learning rule, each output neuron learns to respond with an un-normalized distribution of inputs conditioned on one input class. **(a)** Accuracy of classifying if a pattern originated from the prefrontal cortex or visual cortex, while averaging the responses of the neurons to $k = 1, .., 7$ activity patterns, when both output neurons were properly normalized (using the Wang-Landau algorithm). **(b)** Accuracy of classifying when both output neurons were normalized with a biologically-plausible approximation based on the proportion of silent patterns.

Biologically-plausible training of a neural circuit to discriminate between inputs

An RP circuit with multiple output neurons can classify its inputs by selectively applying the learning rule to its output neurons. Fig. S9 presents a simple proof-of-concept with two identical RP circuits (or, equivalently, a single RP circuit with two output neurons) that can learn to classify whether its inputs have originated from the prefrontal cortex or the visual cortex. The circuits were presented with a mix of activity patterns recorded from these two areas, where the learning rule was applied to output neuron #1 only when input from the visual cortex was presented and to output neuron #2 only when input from the prefrontal cortex was presented. This resulted in the one neuron responding with $-\log p(\vec{x}|\text{visual}) - \log Z_1$ and the other neuron responding with $-\log p(\vec{x}|\text{prefrontal}) - \log Z_2$, where Z_1 and Z_2 are unknown normalization factors.

A maximum-likelihood classifier that classifies a new pattern - $argmax(\vec{x}_{new}|dataset)$ - can be implemented by simply comparing the activation of the two output neurons in response to $\vec{x}_{new}$. Normally, comparing the responses of the same output neuron for two input patterns would result in $\log Z$ canceling out. In this case, when comparing the responses across different output neurons, it is required to find Z_1 and Z_2 (also referred to as the partition functions) and add them as biases to the neurons' responses.

We tried to compensate for the biases $\log Z_1, \log Z_2$ in two different ways:

1. By applying the Wang-Landau algorithm [11], a numerical MCMC-based method for normalizing exponential distributions. This method can give accurate results but has no biologically-plausible implementation.

2. A simple biologically-plausible approximation based on the proportion of time during which the circuit receives no spiking input (see earlier in the SI - "Self-normalization in RP models" - for details).

We applied both of these methods to the output neurons and used them to predict whether input patterns have originated from the visual cortex or the prefrontal cortex. We classified either by using the responses to individual patterns, or by averaging the response across k input patterns. Fig S9 shows the accuracy obtained when classifying activity patterns from groups of 100 input neurons, by applying 10 epochs of the learning rule using 100,000 samples from each of the two datasets, while setting $k = 1, 2..., 7$. With correct normalization (Fig. S9b), classification of the input patterns was near-perfect even using one or two example. With the approximate normalization (Fig. S9c) the accuracy was lower (~90%) when using a single example, but quickly rose when averaging over multiple examples.

Derivation of the noise-based learning rule from Minimum Probability Flow

Minimum Probability Flow [12] is an algorithm for estimating parameters for probabilistic models by minimizing the KL-divergence between the data and the model after running the dynamics for an infinitesimal time ϵ : $\hat{\theta}_{MPF} = \arg\min_\theta D_{KL}\left(p^{(0)}||p^{(\epsilon)}(\theta)\right)$. The authors show that this objective function be approximated by minimizing the objective:

$$K(\vec{\lambda}) = \frac{\epsilon}{|D|} \sum_{x_j \in D} \sum_{x_i \notin D} g_{ij} \exp\left[\frac{1}{2}(E_j(\vec{\lambda}) - E_i(\vec{\lambda}))\right] \tag{S4}$$

where: $x_j \in D$ denote patterns in the training data, $x_i \notin D$ denote patterns not in the training data, and g_{ij} denote elements of a transition-rate matrix Γ which is allowed to be extremely sparse. This is further extended to the case where Γ is sampled rather than deterministic, giving the objective:

$$K(\vec{\lambda}) = \sum_{x_j \in D} \sum_{x_i \notin D} g_{ij} \left(\frac{g_{ji}}{g_{ij}}\right)^{\frac{1}{2}} \exp\left[\frac{1}{2}(E_j(\vec{\lambda}) - E_i(\vec{\lambda}))\right]$$

where the inner sum is obtained by averaging over samples from g_{ij}. Being a convex function, this objective function has a global minimum which can be obtained by iteratively applying its gradient:

$$\frac{\partial K(\vec{\lambda})}{\partial \theta} = \sum_{x_j \in D} \sum_{x_i \notin D} \left[\frac{\partial E_j(\vec{\lambda})}{\partial \vec{\lambda}} - \frac{\partial E_i(\vec{\lambda})}{\partial \vec{\lambda}}\right] g_{ij} \left(\frac{g_{ji}}{g_{ij}}\right)^{\frac{1}{2}} \exp\left[\frac{1}{2}(E_j(\vec{\lambda}) - E_i(\vec{\lambda}))\right] \tag{S5}$$

Minimum Probability Flow's advantage comes from the fact that transition-rate matrix Γ can be very sparse ($g_{ij} = 0$ for most i, j). When this is the case, most of the terms in Equ. S4 and Equ. S5 drop out, and they can be directly computed even for high dimensional input patterns. This solves a major problem in estimating the parameters of maximum entropy models, as directly minimizing the KL-divergence between the model and data requires either summing over all the possible inputs or approximating it with MCMC methods.

Essentially, Eq. S5 sums over pairs of input patterns - those which are part of the training set ($x_j \in D$) and those that are not ($x_i \notin D$), where g_{ij} are elements of a transition matrix that defines the probabilities to switch between states in a Gibbs-style random walk. The learning rule implements a stochastic version of Eq. S5 by taking advantage of three properties:

1. The term $E_j(\vec{\lambda})$ - the energy of state $\vec{x_j}$ - is exactly the membrane potential of the RP output neuron (with synaptic connections $\vec{\lambda}$) in response to the input $\vec{x_j}$.

2. The vector $\frac{\partial E_j(\vec{\lambda})}{\partial \vec{\lambda}}$ is exactly the vector $(h_1(\vec{x_j}), ..., h_k(\vec{x_j}))$ - the response of the RP intermediate neurons to the input $\vec{x_j}$ - because $E_j(\vec{\lambda}) = \sum_i \lambda_i \cdot h_i(x)$.

3. We can create patterns $\vec{x_i} \notin D$ by adding noise to the patterns $\vec{x_j}$: each of the bits in the input pattern is flipped with probability p. Since g_{ij} reflects the probability that $\vec{x_i}$ and $\vec{x_j}$ appear together, this defines a transition matrix Γ where g_{ij} is the probability that pattern $\vec{x_j}$ would be flipped into pattern $\vec{x_i}$:

$$g_{ij} = \begin{cases} (1-p)^n & i = j \\ (1-p)^{(n-k)} p^k & x_i \text{ differs from } x_j \text{ by } k \text{ bits} \end{cases}$$

we denote these noisy patterns as x_{echo}. In particular, $g_{ij} = g_{ji}$ (because bit flipping is symmetric) so $\left(\frac{g_{ji}}{g_{ij}}\right)^{\frac{1}{2}} = 1$. By substituting all of these into equation S5 we obtain:

$$\frac{\partial K(\vec{\lambda})}{\partial \vec{\lambda}} = \sum_{x} \sum_{x_{echo}} \left[h_i(x) - h_i(x_{echo})\right] g_{x,x_{echo}} \exp\left[\frac{1}{2}(E(x) - E(x_{echo}))\right]$$

Because the inner sum is obtained by the random generation of x_{echo} and the outer sum by sampling from the input data, a stochastic gradient descent using this gradient would be equivalent to the learning rule presented in the paper:

$$\frac{\partial K(\vec{\lambda})}{\partial \vec{\lambda}} = \left[h_i(x) - h_i(x_{echo})\right] \exp\left[\frac{1}{2}(E(x) - E(x_{echo}))\right] \tag{S6}$$

Algorithm 1 RP model with pruning and replacement (random selection)

1: Randomly pick a set of k projections $h_1(x)...h_k(x)$ for the model
2: Approximately train the RP model on the empirical data x_{emp}
3: Choose the q projections for which $|\lambda_i|$ is smallest and remove them
4: Generate q new random projections $g_1(x)...g_q(x)$ and add them to the model
5: repeat steps 2-4 until the required amount of projections has been replaced

Algorithm 2 RP model with pruning and replacement (greedy selection)

1: Randomly pick a set of k projections $h_1(x)...h_k(x)$ for the model
2: Approximately train the RP model on the empirical data x_{emp}
3: Choose the q projections for which $|\lambda_i|$ is smallest and remove them
4: Generate a set of r random projections $g_1(x)...g_r(x)$
5: Generate a set of samples $x_{synth} \sim \hat{p}$
6: Add to the model the q projections which maximize: $|\sum_{x_{emp}} g_j(x) - \sum_{x_{synth}} g_j(x)|$
7: repeat steps 2-6 until the required amount of projections has been replaced

References

[1] Ganmor E, Segev R, Schneidman E (2011) Sparse low-order interaction network underlies a highly correlated and learnable neural population code. *Proceedings of the National Academy of Sciences* 108(23):9679–9684.

[2] Turrigiano G (2011) Too many cooks? Intrinsic and synaptic homeostatic mechanisms in cortical circuit refinement. *Annual review of neuroscience* 34:89–103.

[3] Faisal AA, Selen LPJ, Wolpert DM (2008) Noise in the nervous system. *Nature Reviews Neuroscience* 9(4):292–303.

[4] Stuart G, Spruston N, Sakmann B, Hausser M (1997) Action potential initiation and back propagation in neurons of the mammalian central nervous system. *Trends in Neurosciences* 20(3):125–131.

[5] Markram H, Tsodyks M (1997) The neural code between neocortical pyramidal neurons depends on neurotransmitter release probability. *Proceedings of the National Academy of Sciences* 94(2):719–723.

[6] Shahaf G, Marom S (2001) Learning in Networks of Cortical Neurons. *Journal of Neuroscience* 21(22):8782–8788.

[7] Mongillo G, Barak O, Tsodyks M (2008) Synaptic theory of working memory. *Science* 319(5869):1543–1546.

[8] Urbanczik R, Senn W (2014) Learning by the Dendritic Prediction of Somatic Spiking. *Neuron* 81(3):521–528.

[9] Vorster AP, Born J (2015) Sleep and memory in mammals, birds and invertebrates. *Neuroscience and Biobehavioral Reviews* 50:103–119.

[10] Shein-Idelson M, Ondracek JM, Liaw HP, Reiter S, Laurent G (2016) Slow waves, sharp waves, ripples, and REM in sleeping dragons. *Science* 352(6285):590–595.

[11] Wang F, Landau DP (2001) Efficient, multiple-range random walk algorithm to calculate the density of states. *Physical Review Letters* 86(10):2050–2053.

[12] Sohl-Dickstein J, Battaglino PB, Deweese MR (2011) New method for parameter estimation in probabilistic models: Minimum probability flow. *Physical Review Letters* 107(22):11–14.