Task learning through stimulation-induced plasticity in neural networks

Francesco Borra, Simona Cocco, Rémi Monasson

Task learning through stimulation-induced plasticity in neural networks

Francesco Borra, Simona Cocco, and Rémi Monasson
Laboratoire de Physique de l'Ecole Normale Supérieure, PSL Research, CNRS UMR8023, Sorbonne Université, 24 rue Lhomond, 75005 Paris, France

Synaptic plasticity dynamically shapes the connectivity of neural systems and is key to learning processes in the brain. To what extent the mechanisms of plasticity can be exploited to drive a neural network and make it perform some kind of computational task remains unclear. This question, relevant in a bioengineering context, can be formulated as a control problem on a high-dimensional system with strongly constrained and non-linear dynamics. We present a self-contained procedure which, through appropriate spatio-temporal stimulations of the neurons, is able to drive rate-based neural networks with arbitrary initial connectivity towards a desired functional state. We illustrate our approach on two different computational tasks: a non-linear association between multiple input stimulations and activity patterns (representing digit images), and the construction of a continuous attractor encoding a collective variable in a neural population. Our work thus provides a proof of principle for emerging paradigms of *in vitro* computation based on real neurons.

INTRODUCTION

Natural neural networks have long been a key source of inspiration for machine learning and computing, starting from Rosenblatt's perceptron to neuromorphic computing [1] and to nowadays deep learning. Recently, it was suggested that biological neural structures might be directly exploited as a support for *vitro* computation [2, 3]. Progress over the last decade has made it possible to grow, preserve, and study brain organoids [4]. The capabilities of stimulating and recording neural populations [5–7] allow for interfacing these systems, potentially turning brain organoids into miniature biological computers. Computing with organized assemblies of neurons could offer considerable advantages with respect to electronics-based devices, in particular in terms of low energy consumption, massive parallel computation, and continuous learning. However, besides the practical challenges raised by these technologies [8], it remains unclear how they could be best used to achieve high-level computation. The aim of this paper is to address one conceptual question arising in this bio-engineering context: how could circuits of biological neurons be trained to carry out desired computational tasks?

Computation with neural cultures has been largely based so far on the framework of reservoir computing [9, 10], in which the readout of the high-dimensional neural activity is trained to perform the task of interest. However, reservoir computing falls short from fully realizing the computational potential of neural systems, as it does not exploit their capability for reconfiguration of biological connectivity across time. The presence of plasticity mechanisms, involving multiple molecular and cellular processes, is crucial to the formation and maintenance of experience–related changes to neural function and circuitry [11]. These mechanisms can potentially be exploited to shape, through appropriate stimulations, the activity and the connectivity of networks [12], as has been shown in specific settings [13]. From a bio-engineering point of view, the question is therefore to determine spatio-temporal stimulation patterns capable of remodeling a plastic neural network and make it achieve a desired computational task (Fig. 1(a)). Establishing guiding principles and practical tools to obtain those stimulation patterns is essential to future progress in biological computation.

From a mathematical point of view, supervised learning of a task is generally cast as an optimization problem in very high dimensions. Given a population of N (artificial) neurons connected through S synaptic connections J_{ij}, one looks for the minimum of the cost (loss)

function $U\big(\{J_{ij}\}\big)$ expressing the mismatch between the target and actual computation carried out by the network. In artificial nets, the cost U can be gradually reduced through gradient descent (or one of its stochastic variants) in the space of connections, until a minimum with good performance is reached. A key point here is that all moves in the N^2-dimensional interaction space, in particular the one along the gradient of the cost, $\frac{\partial U}{\partial J_{ij}}$, are allowed during learning.

The situation is much more constrained in the case of biological networks, in which the learning dynamics can obviously not be arbitrarily chosen, and plasticity mechanisms set strong limitations about feasible directions. As a concrete example, consider the case of Hebbian-like plasticity rules, in which the changes in the connections ΔJ_{ij} are functions of the firing rates r_i (over some appropriate time scale) of the neurons. As the number N of neurons is generally much smaller than the number N^2 of synaptic interactions, the plastic changes ΔJ_{ij} are highly inter-dependent, and cannot be individually tuned to match the gradient components $\frac{\partial U}{\partial J_{ij}}$. This fundamental limitation makes supervised learning with biologically-plausible rules conceptually much more intricate than with unconstrained dynamics. In addition, the magnitude of synaptic modifications is hard to control in biological networks, while the capability to implement adaptive learning rates is generally regarded as a key ingredient in machine learning.

In the present paper, we show that *in silico* neural networks that obey plausible plasticity mechanisms can be effectively trained, through the application of adequate control stimulations, to carry out very diverse computational tasks. These stimulations vary with time and from neuron to neuron, and are obtained by solving a sequence of optimization problems. Intuitively, the control stimulations are capable of inducing adequate neural activity, progressively driving the network connectivity through its intrinsic Hebbian-like plasticity towards a desired state (Fig. 1(a)). Figure 1(b) summarizes our learning protocol, which relies on multiple cycles of stimulations and recordings of the neural population. At the beginning of each cycle, the responses of the network to few short probing stimulations are recorded, and used to infer the connectivity of the network. Based on this estimate of the connectivity, we plan a control stimulation pattern, which is subsequently applied to the neurons. Under this control, plastic changes to the connections take place and enhance the network performance in achieving the desired computation. The procedure is iterated until the computational target is reached.

MODEL

Dynamics of neural activity

We consider a network of N neurons, characterized by their firing rates $\mathbf{r}(t) = \{r_i(t)\}$ at time t and the time-dependent synaptic connectivity matrix $\mathbf{J}(t) = \{J_{ij}(t)\}$ (Fig. 2). By convention, J_{ij} refers to the coupling from the presynaptic neuron j to the post-synaptic neuron i. The neural population includes N_E excitatory (E) and N_I inhibitory (I) neurons, which constrains the signs of the corresponding synaptic interactions. The activities of the neurons obey the standard dynamical rate equations

$$\tau_n \, \frac{dr_i}{dt}(t) = -r_i(t) + \Phi\left(\sum_j J_{ij}(t)\, r_j(t) + f_i(t)\right) \tag{1}$$

where τ_n is the membrane relaxation time, and $f_i(t)$ is a time-dependent control stimulation on neuron i, which can be dynamically controlled at will in the range $[f_{min}; f_{max}]$. The input-to-rate transfer function Φ is a sigmoidal function, ranging between zero and maximal frequency r_{max} (Methods and SI, Section I). Control stimulations f_i are expressed in units of r_{max}, while connections J_{ij} are dimensionless (SI, Section I).

Synaptic plasticity

Plasticity induces activity-dependent changes in the interactions. While the precise rules describing plasticity are still debated in neuroscience, we use a simple and general model presenting three essential qualitative features: Hebbian associativity, regression towards a baseline (homeostasis), and bounded synaptic strengths. The equation for the synaptic dynamics is

$$\tau_s \frac{dJ_{ij}}{dt}(t) = \underbrace{\eta(\epsilon_j)\,(r_i - \theta(\epsilon_j))\,r_j}_{\text{hebbian}} - \underbrace{\beta_1\,|J_{ij}|\,(r_i^2 - \theta_0(\epsilon_j)^2)}_{\text{homeostasis 1}} - \underbrace{\beta_2\,\text{sign}(J_{ij})\,h_2\big(|J_{ij}| - \bar{J}\big)}_{\text{homeostasis 2}} \quad (2)$$

In the equation above, $\epsilon_j = E$ or I is the pre-synaptic neuron type, and $h_2(u) = u^2$ if $u \geq 0$, 0 if $u < 0$. The first term assumes that changes in the connections derive from a Hebbian-like covariance rule with a post-synaptic threshold depending on the neuron type [14]. The last two terms account for homeostatic feedback [15–18], biasing the activity of the post-synaptic neuron towards a baseline activity θ_0, and imposing a soft clipping of synaptic amplitudes outside the range $[-\bar{J}; +\bar{J}]$. Plasticity for excitatory and inhibitory connections are assumed to have different learning rates $\eta(E)$ and $\eta(I)$ and are associated to different thresholds $\theta(E)$ and $\theta(I)$) to enhance the stability of the network activity states. The parameters β_1 and β_2 control the strengths of the homeostatic constraints.

The learning rule in Eq. (2) is flexible, and can describe Hebbian as well as anti-Hebbian learning. The former is obtained when $\eta(\epsilon_j)$ is positive, independently of the pre-synaptic neuron type ϵ_j. To accommodate for anti-Hebbian inhibition [19], one chooses $\eta(E) > 0, \eta(I) < 0$. We report results with both choices of rules below. We stress that anti-Hebbian here refers to a negative feedback between correlations and synaptic strength change.

While the associative learning rule above can be easily modified, a key assumption we rely on is that synaptic changes are slow compared to the fast variations in the activity, *i.e.* $\tau_n \ll \tau_s$ (Methods and Supplemental Material (SM), Section II). Due to this assumption only the slow variations (on the τ_s time scale) of the stimulations $\mathbf{f}(t) = \{f_i(t)\}$ matter. Furthermore, the neural activities r_i are quasi-stationary and locked to the slowly varying inputs. In mathematical terms, the r.h.s. of Eq. (1) vanishes.

Target state of the network

Our goal is to train the neural network connectivity $\mathbf{J}$ to meet some target, either structural or functional (Fig. 1(a)). In the former case the network connectivity is asked to reach some value, say, $\mathbf{J}^{target}$. The training procedure should ensure that the loss

$$U_{task}(\mathbf{J}) = \sum_{i,j} w_{\epsilon_i,\epsilon_j} \left[J_{ij} - J_{ij}^{target}\right]^2 \quad (3)$$

decays towards zero over the training time. We introduce weights $w_{\epsilon_i,\epsilon_j}$ depending on the neuron types to ensure that the four classes of connections $E, I \to E, I$ are equally contributing to U_{task} (Methods).

In the functional case, the target is not the connectivity itself as above, but the computation carried out by the network. The neurons i in the network are partitioned into three subpopulations, referred to as input (*in*), processing (*proc*), and output (*out*). The network is required to implement a set of n_{pairs} input/output mappings, that is, produce desired activities r_i^μ of the neurons i in the *out* subpopulation in response to specific input stimulations $\mathbf{f}^\mu$ applied to the neurons j in the *in* subpopulation, with $\mu = 1, ..., n_{pairs}$. A possible loss associated to this association task reads

$$U_{task}(\mathbf{J}) = \sum_{\mu=1}^{n_{pairs}} \sum_{i \,\in\, out} \left[r_i(\mathbf{J}, \mathbf{f}^\mu) - r_i^\mu\right]^2 , \quad (4)$$

where $r_i(\mathbf{J}, \mathbf{f})$ refers to the stationary solution of Eq. (1); see Methods for alternative choices

of the loss. The task may be made harder by requiring that the *in* and *out* neurons share no direct connections, and communicate through the neurons in the *proc* subnetwork.

TRAINING LOOP: SUPERVISED CONTROL OF HEBBIAN PLASTICITY

Our training procedure is based on a sequence of learning cycles, labelled by $k = 0, 1, 2, ...$ and sketched in Fig. 2. Each cycle k can be decomposed in three steps:

- Estimation of the current connectivity, $\mathbf{J}_k$, through fast probing of the responses of the network to random stimuli.
- Calculation of the optimal control to be applied, $\mathbf{f}_k^*$, to shift the network connectivity state towards the desired target.
- Application of this control during the period Δt, leading to a reorganization of the network through its intrinsic plasticity mechanisms.

The learning process stops when the value of the loss U_k is considered small enough that the target has been reached. We now give more details about the steps above; technicalities can be found in Methods.

Estimation of the network connectivity

Determining and updating the optimal control stimulation requires knowing, to some degree of accuracy, the state of connectivity $\mathbf{J}_k$ [20]. To mimic realistic conditions, $\mathbf{J}_k$ cannot be accessed through direct measurements. We therefore infer the connectivity based on multiple recordings of the activity (Fig. 2). We probe the network stationary activity states $\mathbf{r}^\nu = \{r_i^\nu\}$ corresponding to various random stimulation patterns $\mathbf{f}^\nu = \{f_i^\nu\}$, where $\nu = 1, ..., n_{probes}$. The connectivity matrix is then estimated so that the stationary equations

$$r_i^\nu = \Phi\left(\sum_j (J_k)_{ij}\, r_j^\nu + f_i^\nu\right) \qquad (5)$$

are fulfilled for all $i = 1, ..., N$ and $\nu = 1, ..., n_{probes}$, see Eq. (1) and Methods. Increasing the number n_{probes} of recordings reduces the error over the estimated connectivity matrix, but the duration of the whole process should be sufficiently short not to induce any plastic modification to $\mathbf{J}_k$: hence n_{probes} should not exceed τ_s/τ_r,where τ_r is the relaxation time of the network activity, which can be estimated from τ_n and $\mathbf{J}_k$ (Methods). In practice, the connectivity varies little after each control stimulation period, and few recordings are needed to update our estimate.

Calculation of optimal control

Ideally, one would like the change of connectivity during one cycle, $\Delta\mathbf{J} = \mathbf{J}_{k+1} - \mathbf{J}_k$, to align along the direction of steepest descent of the loss U, i.e. $\Delta\mathbf{J} \propto -\partial U/\partial\mathbf{J}|_{\mathbf{J}_k}$ However, the connections obey the plasticity rule in Eq. (2) and their dynamics is not directly and arbitrarily controllable. From an informal point of view, the best we can hope for is to find a stimulation that will enhance neuronal activities in such a way as to move connections in a direction as aligned as possible with the gradient of U, see Fig. 3(a).

Let us call $\Delta\mathbf{J}(\mathbf{f}, \Delta t, \mathbf{J}_k)$ the change in the connections produced by applying the stimulation $\mathbf{f}$ for a period of time Δt to the network with initial connectivity state $\mathbf{J}_k$. In principle,

$\Delta\mathbf{J}$ is obtained by integrating the Hebbian-like plasticity rule in Eq. (2) during the time interval (under fixed stimulation). Then, the best control is formally given by

$$\mathbf{f}_k^* = \underset{\mathbf{f}}{\text{argmin}} \quad U\big(\mathbf{J}_k + \Delta\mathbf{J}(\mathbf{f}, \Delta t, \mathbf{J}_k)\big) \ . \quad (6)$$

Solving this optimization problem and finding the absolute minimum can be quite hard. We use a gradient descent procedure in the space of the controls $\mathbf{f}$, not to be confused with the training dynamics over $\mathbf{J}$, which is guaranteed to return a local minimum at least. We stress that this optimization step is an abstract computation, unrelated to any physical or biological process, and is done on a computer separate from the neural model, see Fig. 2. Informally speaking, each cycle can be thought of as an approximate gradient descent of U in the $\mathbf{J}$-space, whose precise direction is determined by another gradient descent in the $\mathbf{f}$-space. This double process is shown in Fig. 3(b).

The determination of $\mathbf{f}_k^*$ in Eq. (6) via gradient descent is a non trivial computational problem due to the non-linearities in the neural and synaptic dynamics, see Methods. Choosing $\Delta t \ll \tau_s$ (SM, Section 1) and imposing additional regularization terms (Methods) ensure that the changes $\Delta\mathbf{J}$ remain small at the end of the learning cycle.

Plastic reorganization of the network under control stimulations

Once the optimal control $\mathbf{f}_k^*$ has been computed (in a time we neglect compared to the time scales at play in the neural network), it is applied to the neurons for a period Δt. The activities of the neurons rapidly adapt to these external stimulations, and the connections start evolving under the plasticity dynamical rules.

At the end of this training cycle, the network connectivity may differ from the expected value $\mathbf{J}_k + \Delta\mathbf{J}(\mathbf{f}_k^*, \Delta t, \mathbf{J}_k)$ due two main factors. First, the estimate $\mathbf{J}_k$, obtained prior to the stimulation, is not equal to the ground truth connectivity. The accuracy depends on the number of measurements, and is limited by the constraint of fast probing as explained above. Second, the true underlying plastic mechanisms controling the network evolution can never be exactly modeled. The unavoidable mismatch between the network dynamics and Eq. (2) induces, at the end of the stimulation period, a systematic bias in the value of the connectivity. To account for this bias and quantify its consequences, we consider below two variants of Eq. (2), with different set of parameters, for the 'true' plastic evolution of the network and for its model counterpart used to compute $\Delta\mathbf{J}(\mathbf{f}, \Delta t, \mathbf{J})$ and the optimal control $\mathbf{f}^*$.

APPLICATIONS

We apply below our learning procedure to two computational tasks; a third task defined on a small network is detailed in SM, Section VI, and in the Discussion section.

Task 1: Non-linear association between all-or-nothing inputs and digit-like outputs

Definition of the task. A fundamental computation carried out by neural circuits in organisms is the association of diverse behavioural (or motor) responses to multiple sensory inputs. This task is particularly non trivial to learn when it is non additive, as it cannot be realized by linear networks [21], and may require to recruit dedicated brain areas [22].

We consider here a toy version of a non-linear associative task, in which a subset of N_{in} input neurons can be stimulated in n_{pairs} distinct ways. Each input stimulation is expected to elicit an associated activity pattern over N_{out} output neurons, see Fig. 4(a). For the sake of visual clarity, the output patterns are represented

as digits, with grey/black pixels corresponding to neurons with low/high activities arranged on a rectangular grid. We make sure that

- the association task is non-linear by carefully choosing the input/output pairs. For instance, the sums of the input stimulations associated to the output digits 0 and 1 is the input required to elicit digit 2; however, 2 is quite distinct from the superimposition of 0 and 1, see Fig. 4(a). As a result of non-linearity the association task cannot be simply implemented through separate neural pathways, activated by one input and silent for the other ones.

- there is no direct connection from the input to the output neurons, see Fig. 4(b). This arrangement mimics the idealized structures of a modular chip. Hence, the information about the input must be processed and conveyed by an intermediate processing (*proc*) region of the network, which contains most of the N neurons. This requirement also ensures that the decrease of the cost function in the initial part of the training is not due to a simple creation of a linear direct input-to-output connection, but rather to the creation of a complex structure.

Training: costs and dynamics. To learn the task, we introduce a cost over the connectivity matrices favoring the desired input/output associations, see Methods, Eq. (20). An additional cost aims at regularizing the connectivity and speeding up convergence towards stationary states of activity, see Methods.

We start from a network with weak, random connections. As our spatio-temporal control stimulation is applied to the neurons (Fig. 4(c)), the connectivity gets modified, and the costs associated to the task and to the regularization decrease, as shown in Fig. 4(d). The activities of neurons progressively cluster into two categories, depending on their expected values in the output patterns (Fig. 4(e)). The learning process is characterized by three stages. First, the activities of all output neurons approximately reach the same low activity level independently of the input pattern, as regularization tends to decrease the amplitudes of the connections and of the activities. Then activities start to separate according to the output patterns. At this stage one can already guess the output digits, see Fig. 4(f). We let the protocol proceed until the two groups (active, inactive) are clearly separated, and the output neuron activities match the target patterns, compare Fig. 4(a)&(f).

Transfer of information and neural representations. How the information is transferred from the input to the output neurons and represented in the *proc* area is investigated in Fig. 5. While the input and output sub-populations in our network both include $N_{in} = N_{out} = 15$ neurons, the associative task effectively takes place in a lower-dimensional space with $n_{pairs} = 4$ dimensions. We show in Fig. 5(a) that learning progressively concentrates most of the $N_{out} = 15$-dimensional activity in the 4-dimensional subspace spanned by the output digits. This dimensionality reduction phenomenon is even stronger when the N_{in} input neurons are stimulated by random combinations of the training inputs associated to the digits, implying the effective creation of a 4-to-4 dimensional channel upon learning. From a circuit point of view, this functional channel is supported by a network of connections, whose histogram is reported in Fig. 5(b). We observe that, during learning, the E and I subpopulations strengthen their connections, with the exception of recurrent interactions within the I neurons that remain weak. The creation of input-specific sub-networks of activity during the learning process can be seen in the animation provided in SI, Section VII A.

We stress that, within this low-dimensional space, the network computation is not linear as imposed by the non-additivity of the task. We show in Fig. 5(c) how the network response to linear combinations of the inputs may largely differ from the superimposition of the responses to the single inputs, see case $0 - 1$ (left). This result confirms that the network is capable of learning complex behaviour beyond the super-

imposition of patterns in the low-dimensional subspace spans by the patterns. On the contrary, in the absence of any specific constraint imposed by the task, computation appears to be approximately linear, see combinations of inputs 0 and 3 producing the 8 digit (right).

Furthermore, we may ask how the network, after training, is able to process and differentiate the patterns outside the input and output sub-populations. We see in Fig. 5(d) that inputs produce the activation of a significant fraction of the neurons in the *proc* area and that many of those neurons respond specifically to distinct inputs, with stronger correlations in the evoked activities for more correlated inputs. In particular, the role of inhibitory neurons is key to the differentiation of input 2 from both 0 and 1, and to depart from additivity as imposed by the task. Further information about how neurons in the *proc* region participates in the association task can be found in SM, Fig. 6.

Task 2: Ring-like connectivity supporting continuous attractor dynamics

Definition of the task. We now aim at reshaping network connectivity into a target matrix $\mathbf{J}^{target}$ through learning. The target connectivity defines a ring attractor, capable of supporting a bump of activity coding for a continuously varying angle. Ring attractors were first theoretically hypothesized [23, 24], and have recently been observed in the ellipsoid body of fly [25]. Neuron connections are organized along two rings, see Fig. 6(a). Excitatory neurons form the outer ring, with connections decays with their distance, while inhibitory neurons compose the smaller inner ring. The connections between the two rings are such that a bump of activity on one side of the outer ring induces inhibition on the diametrically opposite side (see Fig. 6(a)). The position of the bump is arbitrary in the absence of external input, and is otherwise attracted by a weak and localised input to the outer neurons.

Control stimulations are computed with the cost function given by Eq. (3). The time behaviour of the cost is shown in Fig. 6(b) for ten different random initial networks (see Fig. 6(c), left, for a realization of the naive connectivity). After a fast initial decay the cost relaxes to very low values, signalling the success of the learning procedure.

Training: network connectivity and stimulations. Snapshots of the connectivity at different steps of the learning process are displayed in Fig. 6(c). The strong similarity between the network connectivity at the end of learning (compare Figs. 6(a) and (c), step (4)) confers the desired functional properties of continuous attractors to the network. We report in Fig. 6(d) the average firing activities of neurons when a weak input stimulation (intended to pin the bump of activity at a specific angle) is applied. At the initial state of learning (1), the network responds by a weak excess activity simply reflecting the localized pinning input stimulation. Through training, a sub-population of neurons emerges, whose activity is supported by the recurrent connections. This sub-population varies with the angle associated with the input stimulation, see (2) and (3). The emergence of receptive fields is improved with further training cycles (4). Notice that, even when the target structure is very well reconstructed, the direction of the polarization is not perfect, due to tiny correlated defects in the connectivity structure. The presence of these defects does not preclude the accuracy of the coding of angular information in the network [26].

The time behavior of the control stimulations applied to neurons during learning are shown in Fig. 6(e). We observe the presence of long-distance (large angle) correlations at small times, which disappear at the end of the training. To better characterize this angular structure, we plot the Fourier transform (along neuron indices) of the control stimulations in Fig. 6(f). In the initial stage of training, the stimulation is essentially represented by cosine and sine waves, with periods matching the E and I ring extensions. As learning proceeds, higher and higher wave number modes are recruited to refine the short scale structure of the synaptic matrix. This observation is in agree-

ment with the progressive emergence of connectivity in Fig. 6(c).

DISCUSSION

We have shown how appropriate stimulations can induce plastic reconfiguration in a network subject to Hebbian-like plasticity processes and make it capable of accomplishing a prescribed computation. Our approach could successfully be applied to different tasks (see also SM, Section VI) and plasticity rules (Hebbian and anti-Hebbian, see SM, Sections V C and IV C). These results can be thus seen as a proof-of-concept for general-purpose neural training, which would compile a task into a neural network by externally guiding the intrinsic learning mechanisms. At its core, our training method addresses and solves a sort of inverse Hebbian problem: instead of looking for how synaptic interactions change under external inputs, we search for an appropriate set of inputs, or control stimulations implementing desired changes in the interactions through plasticity mechanisms. Notice that, strictly speaking, the issue of modifying connectivity with constrained control is not unique to biological neural networks: neuromorphic circuits, e.g. based on memristors can undergo plastic changes [27], and computation in such networks could be among potential applications of our training protocol. We now discuss our results, both from the computational and the bioengineering/biological points of view.

Computational aspects

Constrained learning and approximate gradient descent. As stressed in the introduction, plasticity rules strongly constrain the manifold of possible changes in the connectivity matrix and straightforward gradient descent of the loss is generally not feasible (Fig. 3(a)). This fundamental limitation is clearly present even in the case of very simple tasks trainable on small networks (Supplemental Material (SM), Section VI and SM, Fig. 7). As the number N of neurons increases, the ratio between the number of control variables (N) and the dimension of the connectivity space (N^2, or a fraction of N^2 depending on network sparsity) diminishes, and the situation ought to become less and less favorable. Fortunately, this effect is counterbalanced by the huge increase in the multiplicity of the networks realizing the task with N [28]. We observe, indeed, that Task 1 is easier to learn when the size N_{proc} of the processing area (Fig. 4(b)) increases, see SM, Section IV A and SM, Fig. 3. As a result, functional targets are easier to reach than structural ones, as the latter cannot benefit from any multiplicity of solutions.

An additional difficulty is the impossibility to use an adaptive learning rate, a key ingredient in the convergence of most machine-learning algorithms. Though some variability in the duration Δt of the control can be considered, this is constrained by biological and experimental time scales and cannot vary over orders of magnitude. As a result, the magnitude of the changes $\mathbf{\Delta J} \propto \Delta t\, \dot{\mathbf{J}}$ in the connectivity after each cycle cannot be tuned at will.

Choice of cost function: structural vs functional targets. The nature of the cost U has also a deep impact on the time course of performance throughout the learning process. Any task could, in principle, be given a structural cost, *e.g.* by running gradient descent of the functional cost on a computer and finding an adequate network $\mathbf{J}^*$ implementing the target task. However, in the structural case, the value of U is not immediately informative about the computation carried out by the network. Good functional performance, such as the ability to create angle-specific receptive fields in Task 2, see Fig. 6(d), may be reached for vanishing U only (SM, Section V C and SM, Fig. 8(a)&(b)), which is difficult to achieve in practice. For this reason, functional costs, for which low values of U directly imply good performances, are preferable. Accordingly, we find that defining U from the worst input-output mismatch, see Methods, Eq. (20), rather than from the average over all pairs as in Eq. (4), gives better performance for Task 1.

A consequence of the multiplicity of solutions

in the functional case is that relatively little changes to the initial (and random) network are generally sufficient to meet the target, compared to what is needed to meet a structural target. This intuition is corroborated by the analysis reported in SM, Section IV E and by the animation provided in SM, Section VII B. We observe that the internal activity of the network remains strongly self-correlated in time during function-based training, providing evidence for the ability of the learning procedure to reach a functional target with moderate changes to the connectivity.

Model uncertainty and robustness. A crucial point to be addressed in future works is the impact of noise and uncertainty. Dynamical noise, both in the neural activity and in the synaptic transmission, could be taken into account through a probabilistic formulation of the control problem. In addition, one should consider measurement errors in the activity recordings, affecting the estimation of the connectivity, or in the stimulation process, leading to inaccurate control of the network.

Taking into account model uncertainty is more delicate. As a matter of fact, neither the equations describing the neural activity dynamics, nor the plasticity rules are exactly known. Any inaccuracy in the model is potentially a source of bias and of correlated errors in the determination of the stimulation protocol. To estimate the impact of such errors, we have studied the performance of our learning scheme in a mismatch setting, in which the plasticity rules governing the evolution of the connectivity and the ones used for the computation of the protocol differed. We show in SM, Section IV B, that successful training is achieved even with 10% (relative) mismatches on the parameters entering the plasticity rules. This result is a good indication of the robustness of our approach to uncertainties in the model. Needless to say, controlling errors in the modelling of the plasticity rules and designing robust stimulation schemes will be crucial for future developments.

Importance of the range of controls. At the implementation level, our training procedure involves high–dimensional optimization over the control stimulation $\mathbf{f}$, and requires careful numerical implementation. A number of parameters in the optimization loop have to be set, whose choice impacts not only the outcome but also the speed of learning, see SM, Sections IV&V. Of crucial importance is the range of variation of the control stimulation applicable to the neurons, $f_{min} \leq f_i \leq f_{max}$. If the range is wide, a lot of variability in the control stimulation schemes is possible, and learning is easy and fast. As $f_{max} - f_{min}$ decreases, the cost decays more slowly (SM, Fig. 3(a)). Further investigations would be needed to better understand if a minimal value of $f_{max} - f_{min}$ exists below which the target task is not reachable any longer and, more generally, how reachability depends on the task. Notice that the protocol we have implemented corresponds to a control strategy with zero time horizon, which optimizes the cost change ΔU at each cycle. Global protocols, derived in the framework of optimal control [29], could possibly achieve higher performance. However, our high-dimensional setting with incomplete knowledge (of connectivity) could suffer from serious error propagation over large time horizons, making implementation difficult.

Continual learning. The final configuration of the network (reached at the end of the control scheme) is in general not a stable state of the learning dynamics and would be lost over time. This is a general issue with systems undergoing continual learning [30]. From a computational point of view, one would either need to re-apply stimulations to maintain the neural circuit in its state or use a cost function that ensures that the final configuration is stable. From a biological point of view, plasticity processes can be downregulated in many ways *in vivo* [11]. Diminishing plasticity is for instance essential at certain stages of brain development [31].

Biological aspects.

A new framework, in vitro *biological computing.* Our work is deeply motivated by the emerging field of *in vitro* computation with

biological neurons. The main approach in this domain so far has been reservoir computing [9, 10, 32–34], which exploits the capability of randomly connected neural populations to produce highly variable patterns of activity, and has been linked to plausible neurobiological mechanisms [35, 36]. Associative effects of plasticity can be exploited for the sake of learning, and reach performance beyond standard reservoir setups [37, 38]. More targeted approaches have been proposed to directly exploit plasticity for task-training, for instance by choosing familiar/unfamiliar inputs as rewards/punishment [3] or by exploiting stimulation avoidance [39]. Our approach, on the contrary, aims at directly controlling and training the neural structure through the determination of adequate stimulations to be imposed to the neurons.

The idea of learning through stimulations is not new to biology, and is the basis for the concept of vaccination. The immune system is endowed with learning mechanisms that allow it to respond to pathogens after appropriate stimulations, e.g. following the administration of a pathogen protein or its coding RNA sequence. The vaccination scheme (timing and dosage) is crucial to the quality of the immunization [40], and can be designed to enhance the production of the so-called broadly neutralizing antibodies having large response spectra [41]. By analogy, one may expect that stimulation protocols optimized to the learning dynamics of neural systems could drive them to desired operational states.

Limitations and plausibility. Our modelling approach is highly simplified and unlikely to capture detailed biological mechanisms, and is primarily intended to explore conceptual challenges in the training of networks with constrained plasticity. Though the learning procedure is robust against mismatches between the 'true' and modelled plasticity mechanisms (SM, Section IV B), it relies on a number of assumptions that need to be discussed in terms of their technological and biological plausibility. First, we have assumed that stimulations could be done at the level of individual neurons: fine stimulation and recording capabilities are being developed [42] but, as of today, optogenetic [5, 43] and electrophysiological [44] techniques rather allow one to target groups of nearby neurons. This limitation could be taken into account in our approach by imposing spatial correlations between the components of $\mathbf{f}$. Second, our protocol could be adapted to constrained controls $\mathbf{f}$, *e.g.* whose components have a given sign (though excitatory and inhibitory tools are available [5, 45, 46]), or discrete (on/off) rather than continuous values. Though these limitations could in principle be enforced, it remains to be seen how they would affect the learning procedure, both in terms of performances and training time.

Separation of neural and plasticity time scales. An important hypothesis is that the time scale τ_r associated to the relaxation of neural activity is much shorted than the duration of the control stimulation during a cycle, Δt, over which the connectivity changes. Under this assumption, the neural activity is stationary, and the estimate of the connectivity can be updated by probing the network response to few random stimuli without inducing synaptic changes (Methods, Eq. (12)). Physiological time scales related to single-neuron dynamics (τ_n in Eq. (1)) are expected to be of the order of tens of milliseconds. It is then crucial that network effects do not slow down the relaxation of the population activity on times $\tau_r \gg \tau_n$, see Methods, Eq. (8). This is ensured in practice in Task 1 through the introduction of a regularization cost on the connectivity, U_{reg}, which prevents the emergence of slow dynamics modes in the network (Methods, Eq. (13)). We have checked that τ_r remains comparable to τ_n throughout the learning process for all the tasks considered here (SM, Section II). While it is difficult to estimate with precision the time scale τ_s associated to plasticity effects in Eq. (2) and, consequently, the duration Δt of control stimulation, a biologically plausible order of magnitude for Δt is of a few tens of seconds (SM, Section I). Note that this is loosely consistent with meaningful plastic modifications happening over few minutes such as shown in [3, 39].

We conclude that the hypothesis $\tau_r \ll \Delta t$ is satisfied. Furthermore, this estimate of Δt would imply that running hundreds of learning cycles would take few hours. We are well aware that these considerations are speculative and indicative of orders of magnitude at best.

Intensities of stimulations. As emphasized above, having a large range of stimulation values at our disposal is crucial for reaching the target task at the end of learning. The values of f_{max} we have employed in Tasks 1 & 2 are a few tens of r_{max}, see Figs. 4(c) and 6(d); details about how to convert input currents into rates can be found in SM, Section I B. Ultrafast optogenetics can elicit spikes in genetically-modified neurons with high probability (for sufficient light pulse intensity) up to hundreds of Hz [47]. Hence, ratios f_{max}/r_{max} close to unity are experimentally accessible. It remains to be seen whether these techniques could be applied to organoids with similar results.

Dependence on plasticity mechanism. From a modelling point of view, we have resorted to a coarse-grained description of activity at the level of firing rates, rather than of spiking events. Both descriptions are compatible with one another, at the plasticity [48–50] and activity [51–53] levels, to the point that artificial spiking networks may be trained by proxy rate-based models [54]. Plasticity at the spike level is a complex process involving multiple timescales, delayed effects and spike train patterns [55], possible co-existing with non-synaptic plasticity [56], different effects depending on the origin of the neuron [57] and meta-plasticity [50]. The coarse-grained plasticity rules we consider respect some general principles: associativity (covariance rule with a threshold, compatible with experiments [58]), locality (dependence on the post and pre-synaptic neurons [57]), and homeostasis [17]. In addition, we test the robustness of our approach with two different plasticity scenarios, based on Hebbian and anti-Hebbian rules for inhibitory synapses in, respectively, Tasks 2 and 3; see SM, Section V C for reverse choices of the rules for inhibitory synapses. In this context, Hebbian and anti-Hebbian learning rules induce, respectively, the weakening and the strengthening of connections between inhibitory neurons with correlated activity, and result in increased or diminished [16] activity. It would be interesting to exhaustively study the performances of our stimulation protocol over a wider class of plasticity rules.

Relevance for neuroscience. As emphasized above, our modelling approach, besides being an oversimplification of the processes taking place in neurons and in their connections, focuses on *in vitro* circuits. Such circuits share many features, at the molecular and cellular levels, with their counterparts in the brain of animals, but also differ in fundamental ways. In particular, our learning protocol, based on control stimulations computed externally, was designed to meet bioengineering contexts and goals, but is *a priori* not directly relevant for neuroscience, e.g. to understand how animals learn in realistic situations.
From this point of view, the interpretation of the training stimulation sequence is an important point to consider. We have observed that, in the case of the structural Task 2, the structure of the training stimulation adapts to the ring symmetry of the target (Figs.˜6(d)&(e)). Briefly speaking, the stimulation protocol engraves Fourier modes on the ring at finer and finer scales as (training) times passes on. This finding shows the deep connection between the spatio-temporal pattern of stimulations and the structure of the target network. Interestingly, the possibility of shaping a network with structured activity patterns is reminiscent of what is speculated to happen during postnatal development, for instance in the visual cortex [59]. It would be interesting to further investigate this analogy,in particular if the activity driving neural development could be interpreted as the result of some sort of control optimization.

METHODS

Network structure and activation function

The fractions of excitatory and inhibitory subpopulations are equal to, respectively, 80%

and 20%. The synaptic structure is encoded in the binary matrix C: neuron j is connected to neuron i if $C_{ij} = 1$, and is not if $C_{ij} = 0$. The structure matrix C is random and fixed during the training, while the intensities J_{ij} of the existing connections ($C_{ij} = 1$) vary over time according to Eq. (2).

The neuron activation Φ is a monotonously increasing function of the input x (which is dimensionally a frequency, see SM) and is chosen to be

$$\Phi(x) = r_{max}\ \frac{\psi(x)}{1+\psi(x)}, \qquad \text{where}$$
$$\psi(x) = \frac{r_0}{r_{max}}\ \log\left[1+\exp\left(\frac{x}{r_0}\right)\right] \quad (7)$$

is a differentiable and invertible sigmoid function. Here, r_{max} is the maximal firing rate reached for large inputs, and r_0 is the magnitude of the spontaneous firing activity in the absence of input (SM, Section I).

Relaxation dynamics of the network activity

While single-neuron activities are associated with the time constant τ_n in Eq. (1), the effective relaxation time τ_r can be longer due to network effects. We estimate τ_r by linearizing the dynamics of the rates around their stationary values, r_i. Denoting by $\mathbf{D}$ the N-dimensional diagonal matrix with elements $D_{ii} = \Phi'(r_i)$ and zero outside the diagonal, we have

$$\tau_r = \frac{\tau_n}{\rho_{min}} \quad , \quad \text{where}$$
$$\rho_{min} = \begin{array}{l}\text{smallest real part}\\ \text{of the eigenvalues of } \mathbf{Id} - \mathbf{D}\cdot\mathbf{J}\end{array}. \quad (8)$$

Estimates of τ_r are shown for the different tasks in SM Fig. 1

Inference of connectivity

* Initialization: estimate of the synaptic structure C and of the neuron types ϵ. The support of the synaptic interactions, $C_{ij} = 0, 1$ and the nature of neurons, $\epsilon_i = E/I$, are initially unknown. To determine their values we start to probe the network activities r_i^ν in response to n_{probe} randomly chosen stimulation patterns f_i^ν, and rewrite the stationary relations in Eq. (5) as

$$\sum_j J_{ij}\ r_j^\nu = t_i^\nu\ , \quad \text{where} \quad t_i^\nu = \Phi^{-1}(r_i^\nu) - f_i^\nu \quad (9)$$

for all i, ν. As the initial number of probing stimulations does not have to be small (plastic changes would simply affect the starting state of the connectivity for the training phase), we can choose $n_{probe} \geq N$, and the linear system above generically determines the matrix $\mathbf{J}$ in a unique way. To avoid large numerical errors on currents through Φ^{-1}, the probing stimulations f_i^ν are chosen to be large enough to avoid the presence of low firing rates ($r \simeq 0$), see SM, Section I.

Once J has been estimated, we estimate the identity of neurons and the support of the connectivity through, respectively,

$$\epsilon_i = \begin{cases} E & \text{if} \quad \sum_j J_{ji} > 0 \\ I & \text{otherwise} \end{cases} \quad (10)$$

and

$$C_{ij} = \begin{cases} 1 & \text{if} \quad |J_{ij}| > J_{min} \\ 0 & \text{otherwise} \end{cases}\ , \quad (11)$$

where J_{min} is a small threshold, set to 10^{-6}. These estimates of ϵ and $\mathbf{C}$ are left unchanged throughout the training process.

* Updating the connectivity estimate after a training period. Let $\mathbf{J}_i$ be the vector of connections incoming onto neuron i prior to a training stimulation period of duration Δt; the dimension of $\mathbf{J}_i$ is equal to $d_i = \sum_j C_{ij}$. After the training period the vector connections is $\mathbf{J}'_i$. To estimate this vector we probe the network with n_{probes} stimuli, and record the corresponding activities. We call $\mathbf{R}$ the $(n_{probes} \times d_i)$-dimensional matrix of activities r_j^ν and $\mathbf{t}_i$ the n_{probes}-dimensional vector of components t_i^ν appearing in Eq. (9). The solution to the constraints $\mathbf{R}\cdot\mathbf{J}'_i = \mathbf{t}_i$ is therefore not unique. To

lift this ambiguity we select the solution $\mathbf{J}'_i$ closest (in terms of L_2 norm) to the previous estimate $\mathbf{J}_i$. Formally, we obtain [60]

$$\mathbf{J}'_i = \mathbf{\Pi}_i^{\perp} \cdot \mathbf{J}_i + \mathbf{R}^+ \cdot \mathbf{t}_i \ , \tag{12}$$

where $\mathbf{R}^+$ denotes the pseudo-inverse of $\mathbf{R}$ and $\mathbf{\Pi}_i^{\perp} = \mathrm{Id} - \mathbf{R}^+ \cdot \mathbf{R}$ is the projector orthogonal to the n_{probes}-dimensional space spanned by the rows of $\mathbf{R}$. This updating step is then iterated over the N neurons i to obtain all the vectors $\mathbf{J}'_i$.

Regularization of the connectivity matrix

A common practice in machine learning is to introduce a regularization cost over the model parameters, hereafter denoted as $U_{reg}(\mathbf{J})$, to smooth out the training trajectories. In addition, regularization can help ensure that the neural activities reach their stationary values quickly (under fixed stimulation) through the network dynamics in Eq. (1), which is important for training and for connectivity estimation.

We impose regularization to bound the modulus $\sigma(\mathbf{J})$ of the largest singular value of the connectivity matrix. The rationale is that a sufficient condition for the network dynamics to have a unique activity stationary state and converge exponentially fast to this unique state is that $\sigma(\mathbf{J}) < 1$, see Eq. (8) and SI, Section 2.2. For each singular value λ_k of $\mathbf{J}$, the regularization cost is approximately zero if $\lambda_k \ll 1$, and grows linearly as $g_2\,\lambda_k$ for $\lambda_k \gg 1$. We implement this dependence through a softplus function:

$$U_{reg}(\mathbf{J}) = \frac{g_2}{g_1} \sum_{k=1}^{N^2} \log\left(1 + e^{\,g_1\,(\lambda_k - 1)}\right) \tag{13}$$

with $g_2 = 2,\ g_1 = 10$. This regularization cost can be explicitly differentiated with respect to the entries of the connectivity matrix, see below.

Determination of the optimal stimulation $\mathbf{f}^*$

* Derivative of the costs with respect to connections and time. The task cost U_{task} depends on the connectivity $\mathbf{J}$, both directly and indirectly through the stationary activity $\mathbf{r}$. The expression for the total derivative of U_{task} with respect to an entry of $\mathbf{J}$ is, according to chain rule,

$$\frac{dU_{task}}{dJ_{ij}} = \frac{\partial U_{task}}{\partial J_{ij}} + \sum_k \frac{\partial U_{task}}{\partial r_k} \frac{\partial r_k}{\partial J_{ij}} \tag{14}$$

which requires knowledge of the derivatives of the neuron firing rates. These can be computed using the implicit function theorem [61] applied to the stationary equations for the dynamics, with the result:

$$\frac{\partial r_k}{\partial J_{ij}} = \left[\left(\mathbf{Id} - \mathbf{D} \cdot \mathbf{J}\right)^{-1}\right]_{k,i} \mathbf{D}_{ii}\, r_j \ , \tag{15}$$

where $\mathbf{D}$ was first used in Eq. (8). The gradient of the regularization cost, U_{reg} in Eq. (13), can be easily expressed using the derivatives of the singular values: $\frac{d\lambda_k}{dJ_{ij}} = u_{k,i}\, v_{k,j}$, where $\mathbf{u}_k$ and $\mathbf{v}_k$ denote, respectively, the left and right eigenvectors of $\mathbf{J}$ associated to λ_k.

In addition, the gradient component associated to the connection J_{ij} is set to zero if $C_{ij} = 0$ (no connection is actually present), or if $J_{ij} = 0$ and descending the gradient would violate the constraint on the sign of the interaction resulting from the pre-synaptic neuron type, *i.e.* $\frac{dU}{dJ_{ij}} > 0$ if $\epsilon_j = E$ or $\frac{dU}{dJ_{ij}} < 0$ if $\epsilon_j = I$. Notice that knowledge of the gradients of U_{task} and U_{reg} with respect to J_{ij} gives access to the time derivatives of these costs (under fixed stimulation) through the generic formula

$$\dot{U} = \sum_{i,j} \frac{\partial U}{\partial J_{ij}}\, \dot{J}_{ij} \ , \tag{16}$$

where the latter term can be directly obtained from the dynamics over synapses described by Eq. (2).

* Objective function and gradient over stimulations. The total cost U is defined as

the sum of the task and regularization costs, $U = U_{task} + U_{reg}$. To determine the best control stimulation $\mathbf{f}$ over the next period we search for the minimum of the modified change

$$\Delta U(\mathbf{f};\gamma) = U\big(\mathbf{J}+\Delta\mathbf{J}(\mathbf{f})\big) - U\big(\mathbf{J}\big) + \Delta t\ \gamma\ \|\Delta\mathbf{J}(\mathbf{f})\|^2 \tag{17}$$

where $\mathbf{J}$ is the matrix of connectivity estimated through probing at the end of the previous control stimulation period, $\gamma = 1/((\bar{J}/10)^2\, c\, N^2)$ and

$$\Delta\mathbf{J}(\mathbf{f}) = [\mathbf{J} + \Delta t\, \dot{\mathbf{J}}(\mathbf{f})]_{\boldsymbol{\epsilon}} - \mathbf{J} \tag{18}$$

where $[\cdot]_{\boldsymbol{\epsilon}}$ indicates that connections have been clipped to respect their excitatory/inhibitory types ϵ imposed by the pre-synaptic neurons. The last term in Eq. (17) constrains the change in the interactions to be small over the period of stimulation, effectively introducing a saturating non-linearity over the gradients.

Computing the gradient of ΔU in Eq. (17) with respect to the control stimulation components f_i requires the expressions for the derivatives of the costs with respect to the connections in Eqs. (14),(15), and with respect to time, see Eq. (16). In addition, the derivatives of the firing activities are needed. Using again the implicit function theorem we obtain, see Eq. (15),

$$\frac{\partial r_k}{\partial f_i} = \left[\left(\mathbf{Id} - \mathbf{D}\cdot\mathbf{J}\right)^{-1}\right]_{k,i} \mathbf{D}_{ii}\ . \tag{19}$$

To enforce the conditions $f_{min} < f_i < f_{max}$ we set the gradient components of ΔU_{tot} to zero when these boundaries are met.

* Minimization of the cost. The optimal control $\mathbf{f}^*$ can be found through an iterative descent of ΔU in Eq. (17). The control is updated along minus the gradient of $\Delta U(\mathbf{f},\gamma)$, and the resulting f_i's are clipped if the boundaries f_{min} or f_{max} are crossed. The iterative process halts if the decrease of $\Delta U(\mathbf{f},\gamma)$ is smaller (in absolute value) than some threshold, or the expected decrease (based on a fit of the previous steps in the process) is too low. Then, if $\Delta U(\mathbf{f},0)$ is negative (i.e. the cost function is decreasing), the solution is accepted, while the gradient descent is repeated otherwise.

Details about the implementation, with the pseudocode for the optimization loop and the values of the hyper-parameters involved can be found in SM, section III.

Task 1

* Initialization and setting. Elements of the structure matrix C_{ij} are randomly set to 0 or 1 with respective probabilities $1 - c_E$ and c_E for excitatory columns (J such that $\epsilon_j = E$) and $1 - c_I$ and c_I for inhibitory ones ($\epsilon_j = I$). To ensure that the task is not easily implementable no direct connection between the N_{in} input neurons and the N_{out} output neurons is present.Initial connections J_{ij} are drawn uniformly at random in the ranges $[0, J_0]$ if $\epsilon_j > 0$ or $[-J_0, 0]$ if $\epsilon_j < 0$. See SM, Section I for parameter values.

* Task-associated cost. We define a set of n_{task} input stimulations $\mathbf{f}^\mu$ (defined over the N_i input neurons) and output binary activation $\boldsymbol{\sigma}^\mu$ (over the N_o output neurons), with $\mu = 1, ..., n_{task}$. The goal of training is that output neuron i should display High or Low firing activity in response to input μ when, respectively, $\sigma_i^\mu = H$ or L. In practice we enforce that the firing rates associated to neurons with low ($\sigma = L$) and high ($\sigma = H$) activities should differ by more than a prescribed gap $\delta r = 0.12\, r_{max}$. The task cost reads

$$U_{task}(\mathbf{J}) = \frac{\displaystyle\sum_{(\mu,i):\sigma_i^\mu=L}\ \sum_{(\nu,j):\sigma_j^\nu=H} \Delta(\mu,i;\nu,j)\ e^{\gamma\,\Delta(\mu,i;\nu,j)}}{\displaystyle\sum_{(\mu,i):\sigma_i^\mu=L}\ \sum_{(\nu,j):\sigma_j^\nu=H} e^{\gamma\,\Delta(\mu,i;\nu,j)}} \tag{20}$$

where $\gamma = 1/(\delta r/2)^2$ and we have introduced the mismatch

$$\Delta(\mu, i; \nu, j) = h_2\big(r_i^\mu - r_j^\nu + \delta r\big) \tag{21}$$

with

$$h_2(u) = u^2 \text{ if } u > 0,\ 0 \text{ otherwise.} \tag{22}$$

The exponential factors give more weights to large mismatches Δ in the expression of the cost. The dependence of the cost on $\mathbf{J}$ (and on the input stimulations $\mathbf{f}^\mu$) in Eq. (21) is implicit through the firing rates r_i. Minimizing U_{task} is thus equivalent to separating the activities of the H and L neurons as required by the classification task, *i.e.* making all $\Delta(\mu, i; \nu, j) = 0$.

* Analysis of input and output spaces.

To characterize how the association mechanism emerges across training, we convert digits σ_i^μ =High/Low to 1/0 valued arrays. These digits span a $n_{pairs} = 4$-dimensional space. Given a stimulation $\mathbf{f}$, we call $\mathbf{r}_{out}(\mathbf{f})$ the output activity and

$$\|\mathbf{r}_{out}^\perp(\mathbf{f})\|^2 = \frac{\min_{\boldsymbol{\beta}} \left\| \sum_{\mu=1}^{n_{task}} \boldsymbol{\sigma}_\mu^i \beta_\mu - \mathbf{r}_{out}(\mathbf{f}) \right\|^2}{\|\mathbf{r}_{out}(\mathbf{f})\|^2} \ . \tag{23}$$

the normalized squared length of its projection orthogonal to the digit space. In Fig. 5(a), we report this quantity for: (1) generic inputs $\mathbf{f}$ with 0,1 entries with 50-50 probability, and (2) random, uniform convex combinations of the inputs associated to the digits: $\mathbf{f} = \sum_{\mu=1}^{n_{task}} \alpha_\mu \mathbf{f}_\mu$ with $\alpha_\mu \geq 0$ and $\sum_\mu \alpha_\mu = 1$.

Task 2

* Initialization and setting. The connectivity matrix is initialized as in the classification task above, but no *a priori* restriction is imposed to the structure, *i.e.* $c_E = c_I = 1$. See SM, Section I for parameter values.

* Task and cost. The structural cost is defined in Eq. (3). Weights are defined as $w_{\epsilon_i,\epsilon_j} = \big(\sum_{ab} [J^{target}_{\epsilon_i\epsilon_j}]^2_{ab}\big)^{-1}$, where $J^{target}_{\epsilon_i\epsilon_j}$ is the submatrix of $\mathbf{J}^{target}$ connecting neurons of type $\epsilon_j = E/I$ to neurons of type $\epsilon_i = E/I$. This re-weighting ensures that large, *e.g.* $E \to E$, and small, *e.g.* $I \to I$ blocks equally contribute to the cost, and are simultaneously and properly learned during the training process.

The target connectivity is defined as follows. Neurons are associated angles on two rings, depending on their types $\epsilon = E$ or I. The angle attached to the excitatory neuron i is $\theta_i^E = 2\,\pi\,\frac{i}{N_E}$, where N_E is the number of excitatory neurons. A similar formula holds for the N_I inhibitory neurons: $\theta_i^I = 2\,\pi\,\frac{i}{N_I}$. The target synaptic connection from neuron j to i reads

$$J_{ij}^{target} = J_{\epsilon_i,\epsilon_j} \times \mathcal{M}\big(\theta_i^{\epsilon_i} - \theta_j^{\epsilon_j} - \pi\,\delta_{\epsilon_i,I}; K_{\epsilon_i,\epsilon_j}\big) \ , \tag{24}$$

where $\delta_{\cdot,\cdot}$ denotes the Kronecker delta. The angular modulation is done through von Mises function,

$$\mathcal{M}(\Delta\theta; K) = e^{-K + K\,\cos(\Delta\theta)} \ , \tag{25}$$

which is maximal for vanishing angular separation ($\Delta\theta = 0$) and decays over the width $\simeq K^{-1/2}$. Parameter values are listed in SM, Section I.

Receptive fields. We apply a weak local input stimulation $f_i = 0.06\,r_{max}$, $f_{i\pm1} = 0.028\,r_{max}$, $f_{i\pm2} = 0.012\,r_{max}$, $f_{i\pm3} = 0.004\,r_{max}$ to induce polarization of neuron i, and repeat the process for all i. Results for the activities are averaged over 100 trials.

[1] S. Furber, Journal of neural engineering **13**, 051001 (2016).

[2] L. Smirnova, B. S. Caffo, D. H. Gracias, Q. Huang, I. E. Morales Pantoja, B. Tang, D. J. Zack, C. A. Berlinicke, J. L. Boyd, T. D. Har-

ris, E. C. Johnson, B. J. Kagan, J. Kahn, A. R. Muotri, B. L. Paulhamus, J. C. Schwamborn, J. Plotkin, A. S. Szalay, J. T. Vogelstein, P. F. Worley, and T. Hartung, Frontiers in Science **1** (2023).
[3] B. J. Kagan, A. C. Kitchen, N. T. Tran, F. Habibollahi, M. Khajehnejad, B. J. Parker, A. Bhat, B. Rollo, A. Razi, and K. J. Friston, Neuron **110**, 3952 (2022).
[4] M. Lancaster, M. Renner, C. Martin, D. Wenzel, L. Bicknell, M. Hurles, T. Homfray, J. Penninger, A. Jackson, and J. Knoblich, Nature **501**, 373 (2013).
[5] K. Deisseroth, Nature Neuroscience **18**, 1213 (2015).
[6] W. Yang and R. Yuste, Current Opinion in Neurobiology **50**, 211 (2018).
[7] H. Shin, S. Jeong, J. Lee, and al, Nat. Commun. **12**, 492 (2021).
[8] X. Qian, H. Song, and G.-l. Ming, Development **146**, dev166074 (2019).
[9] G. Tanaka, T. Yamane, J. B. Héroux, R. Nakane, N. Kanazawa, S. Takeda, H. Numata, D. Nakano, and A. Hirose, Neural Networks **115**, 100 (2019).
[10] T. Sumi, H. Yamamoto, Y. Katori, K. Ito, S. Moriya, T. Konno, S. Sato, and A. Hirano-Iwata, Proceedings of the National Academy of Sciences **120**, e2217008120 (2023).
[11] A. Citri and R. Malenka, Neuropsychopharmacol. **33**, 18 (2007).
[12] L. Carrillo-Reid, W. Yang, J.-e. Kang Miller, D. S. Peterka, and R. Yuste, Annual Review of Biophysics **46**, 271 (2017).
[13] B. Liu, M. J. Seay, and D. V. Buonomano, Journal of Neuroscience **43**, 82 (2023).
[14] F. J. Vico and J. Jerez, Neural Processing Letters **18**, 1 (2003).
[15] G. G. Turrigiano, Cell **135**, 422 (2008).
[16] G. G. Turrigiano and S. B. Nelson, Current Opinion in Neurobiology **10**, 358 (2000).
[17] F. Zenke, W. Gerstner, and S. Ganguli, Current opinion in neurobiology **43**, 166 (2017).
[18] J. Zierenberg, J. Wilting, and V. Priesemann, Physical Review X **8**, 031018 (2018).
[19] P. Földiak, Biol. Cyb. **64** (1990).
[20] I. M. de Abril, J. Yoshimoto, and K. Doya, Neural Networks **102**, 120 (2018).
[21] M. Minsky and S. Papert, *Perceptrons: An Introduction to Computational Geometry* (MIT Press, Cambridge, MA, USA, 1969).
[22] J. Devaud, T. Papouin, J. Carcaud, J. Sandoz, B. Grünewald, and M. Giurfa, Proceedings of the National Academy of Sciences (USA) **112**, E5854–E5862 (2015).
[23] S. Amari, Biol. Cybern. **27**, 77 (1977).
[24] R. Ben-Yishai, R. Bar-Or, and H. Sompolinsky, Proc Natl Acad Sci USA **25**, 3844 (1995).
[25] J. D. Seelig and V. Jayaraman, Nature **521**, 186–191 (2015).
[26] T. Kuehn and R. Monasson, Physical Review E (2023).
[27] T. Serrano-Gotarredona, T. Masquelier, T. Prodromakis, G. Indiveri, and B. Linares-Barranco, Frontiers in neuroscience **7**, 2 (2013).
[28] E. Marder and A. Taylor, Nat Neurosci **14**, 133–138 (2011).
[29] D. E. Kirk, *Optimal control theory: an introduction* (Courier Corporation, 2004).
[30] L. Wang, X. Zhang, H. Su, and J. Zhu, IEEE Transactions on Pattern Analysis & amp; Machine Intelligence **46**, 5362 (2024).
[31] K. Carstens and S. Dudek, Current Opinion in Neurobiology **54**, 194 (2019).
[32] M. Lukoševičius and H. Jaeger, Computer science review **3**, 127 (2009).
[33] J. B. George, G. M. Abraham, K. Singh, S. M. Ankolekar, B. Amrutur, and S. K. Sikdar, Biosystems **126**, 1 (2014).
[34] Y. Yada, S. Yasuda, and H. Takahashi, Applied Physics Letters **119** (2021).
[35] M. Rigotti, D. B. D. Rubin, X.-J. Wang, and S. Fusi, Frontiers in computational neuroscience **4**, 24 (2010).
[36] P. Enel, E. Procyk, R. Quilodran, and P. F. Dominey, PLoS computational biology **12**, e1004967 (2016).
[37] G. B. Morales, C. R. Mirasso, and M. C. Soriano, Neurocomputing **461**, 705 (2021).
[38] H. Cai, Z. Ao, C. Tian, Z. Wu, H. Liu, J. Tchieu, M. Gu, K. Mackie, and F. Guo, bioRxiv , 2023 (2023).
[39] A. Masumori, N. Maruyama, L. Sinapayen, T. Mita, U. Frey, D. Bakkum, H. Takahashi, and T. Ikegami, in *Artificial Life Conference Proceedings* (MIT Press, Cambridge, 2015) pp. 373–380.
[40] M. Molari, K. Eyer, J. Baudry, S. Cocco, and R. Monasson, eLife **9**, e55678 (2020).
[41] K. Sprenger, J. Louveau, P. Murugan, and A. Chakraborty, Proceedings of the National Academy of Sciences **117**, 20077 (2020).
[42] H. Adesnik and L. Abdeladim, Nature neuroscience **24**, 1356 (2021).

[43] A. M. Stamatakis, M. J. Schachter, S. Gulati, K. T. Zitelli, S. Malanowski, A. Tajik, C. Fritz, M. Trulson, and S. L. Otte, Frontiers in neuroscience , 496 (2018).
[44] M. E. Spira and A. Hai, Nature nanotechnology **8**, 83 (2013).
[45] B. B. Land, C. E. Brayton, K. E. Furman, Z. LaPalombara, and R. J. DiLeone, Frontiers in behavioral neuroscience **8**, 108 (2014).
[46] X. Han and E. S. Boyden, PloS one **2**, e299 (2007).
[47] T. Mager, D. Lopez de la Morena, V. Senn, and al., Nat. Commun. **9**, 1750 (2018).
[48] E. M. Izhikevich and N. S. Desai, Neural computation **15**, 1511 (2003).
[49] D. Bush, A. Philippides, P. Husbands, and M. O'Shea, Neural computation **22**, 2059 (2010).
[50] P. Yger and M. Gilson, Frontiers in computational neuroscience **9**, 138 (2015).
[51] S. Ostojic and N. Brunel, PLoS computational biology **7**, e1001056 (2011).
[52] B. Rueckauer and S.-C. Liu, in *2018 IEEE international symposium on circuits and systems (ISCAS)* (IEEE, 2018) pp. 1–5.
[53] A. Sanzeni, M. H. Histed, and N. Brunel, PLoS computational biology **16**, e1008165 (2020).
[54] S. R. Kheradpisheh, M. Mirsadeghi, and T. Masquelier, IEEE Access **10**, 70769 (2022).
[55] R. C. Froemke and Y. Dan, Nature **416**, 433 (2002).
[56] R. Mozzachiodi and J. H. Byrne, Trends in neurosciences **33**, 17 (2010).
[57] N. Caporale and Y. Dan, Annu. Rev. Neurosci. **31**, 25 (2008).
[58] S. Lim, J. L. McKee, L. Woloszyn, Y. Amit, D. J. Freedman, D. L. Sheinberg, and N. Brunel, Nature neuroscience **18**, 1804 (2015).
[59] C. P. Danka Mohammed and R. Khalil, Frontiers in systems neuroscience **14**, 29 (2020).
[60] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to linear regression analysis* (John Wiley & Sons, 2021).
[61] W. Rudin, *Principles of mathematical analysis* (McGraw-Hill, 1953).

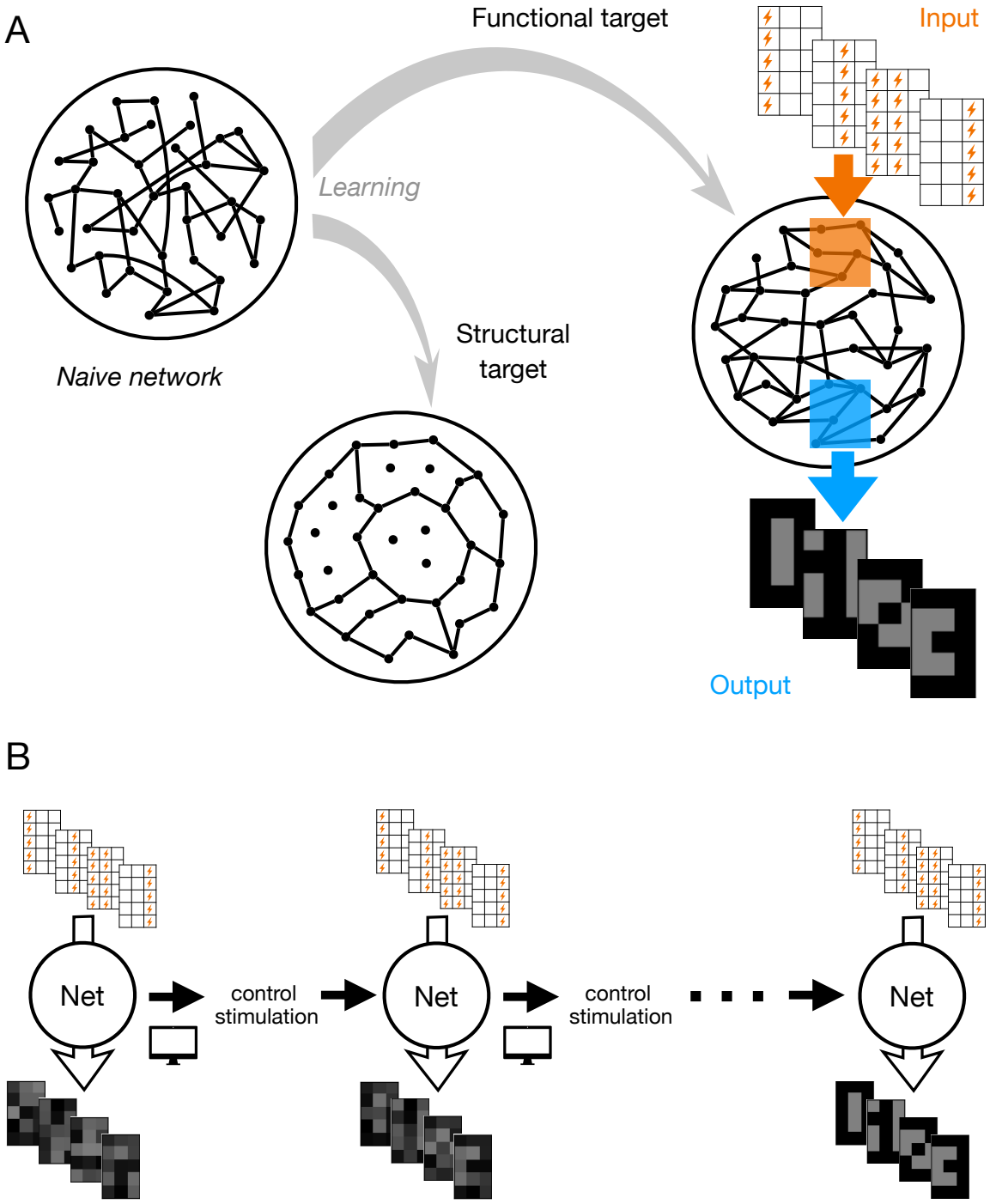

FIG. 1. **Computational targets for plastic neural networks.**
(a) Our goal is to reconfigure a naive neural network and reach some target, either structural (a specific connectivity state) or functional (for instance, the network is required to implement some input–output associations). This reshaping is achieved through a learning process, in which appropriate spatio-temporal stimulations of the neurons exploit intrinsic plasticity mechanisms.
(b) The computational target, here, the functional task in panel (a), is reached through a learning cycle. Prior to the learning process, the naive network associates the inputs to random outputs (left). Our algorithm computes the best control to be applied to the network to modify its connectivity through plastic changes. Applying this control stimulation to the neural population results in an enhancement of the network performance (middle). The control is then re-optimized and applied during a new stimulation period. After multiple iterations, the correct input-to-output association is achieved (right).

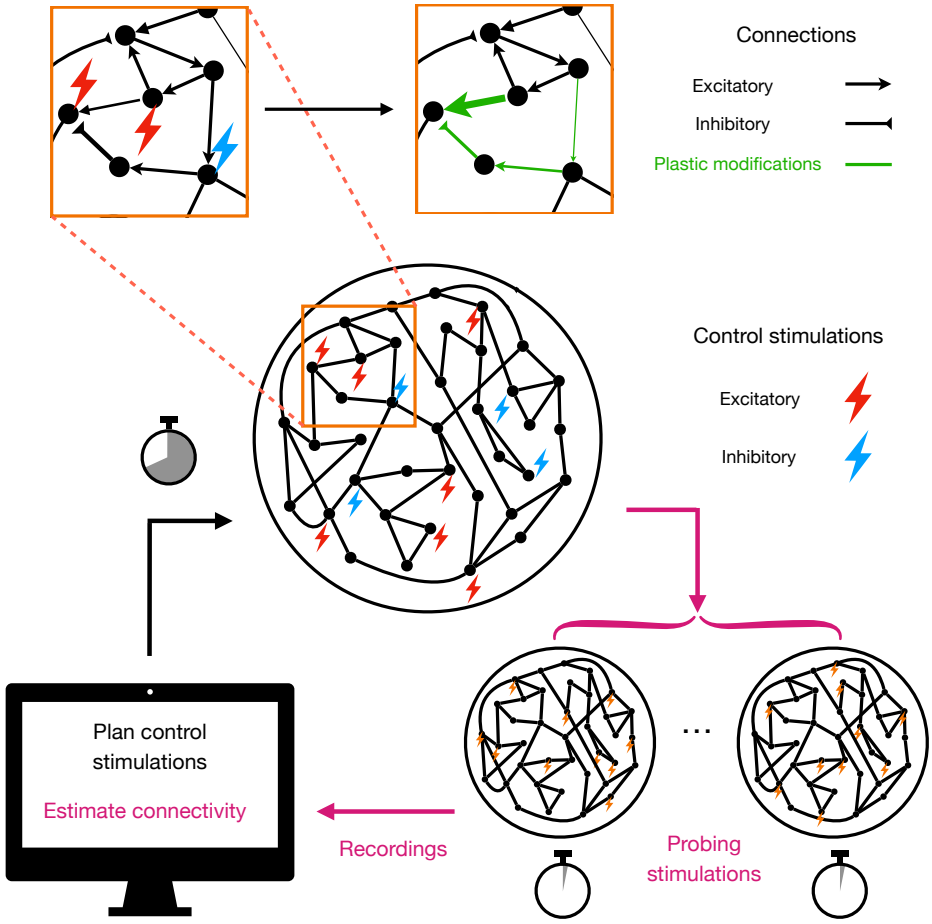

FIG. 2. **Stages of the learning cycle: planning of control stimulations, reconfiguration under plasticity effects, and connectivity estimation**
In each cycle, a control stimulation f_i^* is applied to the neurons i in the network, generating stationary firing rates r_i (middle). In turn this activity pattern leads, through the plasticity rule, to specific strengthening or weakening of the connections J_{ij}, indicated by the changes in the thicknesses of the connection arrows (top). Once the control period halts, few short and random stimulations are applied to the neurons, and the corresponding activities are recorded (bottom, right). These data are then used to update the estimate of the network connectivity, and to plan the optimal control for the next cycle (bottom, left).

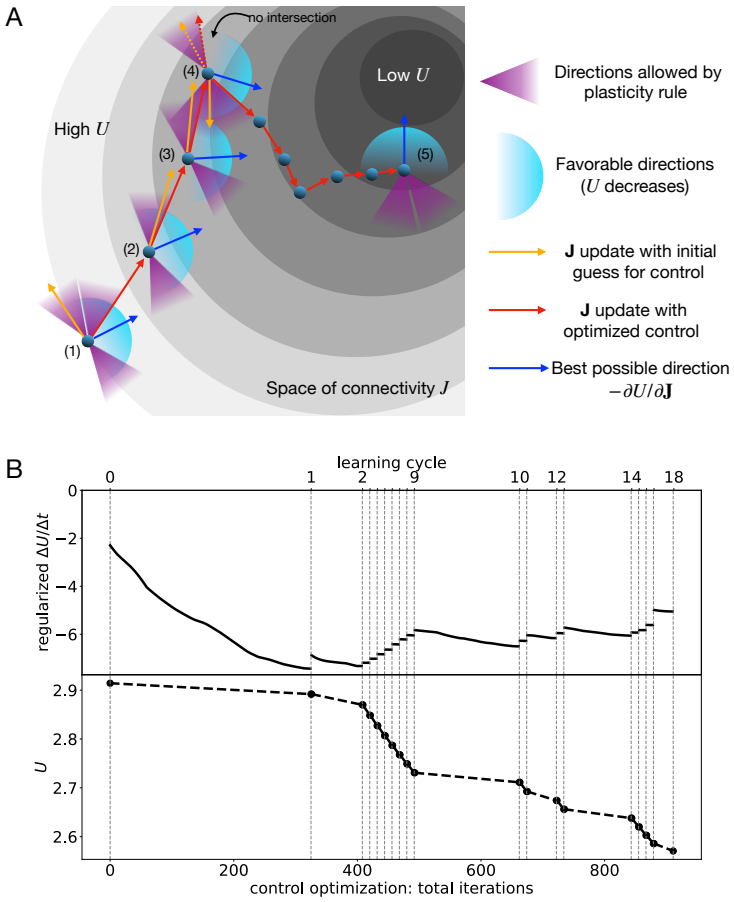

FIG. 3. **Schematic dynamics of the network and feasible directions in the high-dimensional connectivity space during the training process.**

(a) Grey levels show values of the cost U, which quantifies the mismatch between the current connectivity $\mathbf{J}$ of the network and the target. Dark blue balls locate the connectivity $\mathbf{J}$ at the beginning of each control stimulation period. The set of all directions in the N^2-dimensional connection space along which U decreases is shown by the cyan area, centered around the direction of steepest descent of U indicated by the blue arrows. The purple cone symbolically represents the set of all feasible directions for $\mathbf{J}$ under the plasticity dynamics, *i.e.* which can be reached under any N-dimensional control stimulation $\mathbf{f}$.

(1) Initial situation, prior to any control stimulation. The synaptic change corresponding to a random $\mathbf{f}$ may point to a 'bad' direction outside the cyan region (orange arrow), while the change associated to the optimal control stimulation $\mathbf{f}^*$ lies on the edge of the purple cone (red arrow), as close as possible to the best direction (blue arrow).

(2) After one control stimulation period. Repeating the same control stimulation as before would lead to a suboptimal change of $\mathbf{J}$ (orange arrow), while the best stimulation (red arrow) yields a larger decrease in U.

(3) Control stimulations are updated to ensure optimal synaptic changes until **(4)** all local updates of the previous $\mathbf{f}^*$ lie outside the 'good' (cyan) region. We then choose a new initial $\mathbf{f}$, and resumes the search for the optimal $\mathbf{f}^*$ as in cycle (1). This process is iterated, until the change in U are very small and the optimization is completed, or the feasible and 'good' regions do not overlap any longer (point **(5)**). The performance of the network (final value of U) is then assessed.

(b) Decay rate of the loss during the optimization over the control (top) and resulting loss U_k after each learning cycle k (bottom), for the setting corresponding to Task 1. Abscissa indicates the number n of optimization steps. Vertical dashed lines locate the beginnings of the training cycles k. Top: decay rate $(U(n) - U_k)/\Delta t$ at step n of the optimization algorithm, calculated with the regularized loss, see Eq. (17). For each cycle, the rate is decreased through gradient descent over $\mathbf{f}$ until a plateau is reached. The optimization then halts, and the control defines $\mathbf{f}_k^*$. At the begining of the new cycle, the network connectivity $\mathbf{J}$ has changed, the previous control is not optimal any longer, and the optimization process resumes. The plateau values of the rate increase over cycles, making progressively harder to find a good control to train the network.

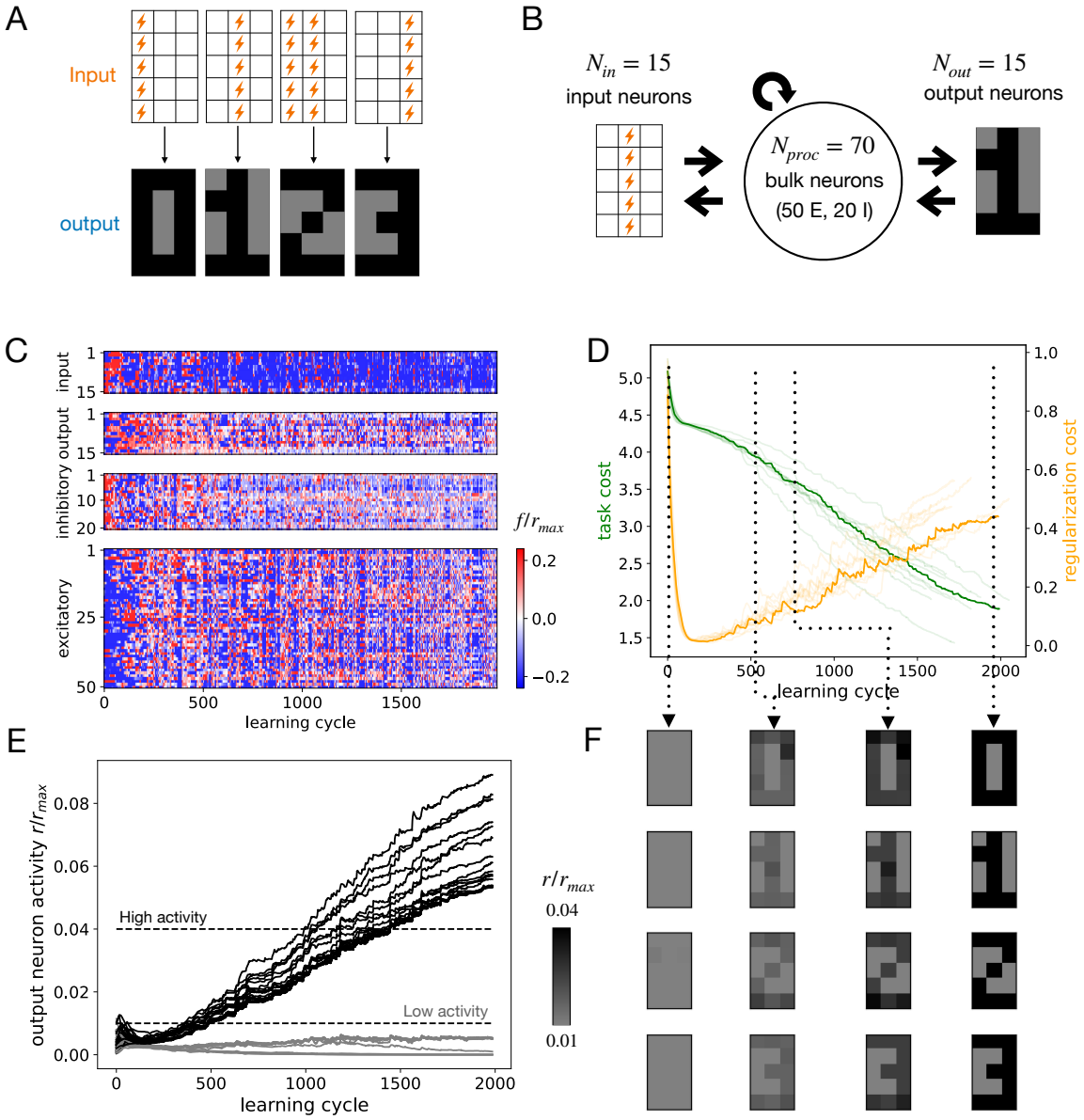

FIG. 4. **Task 1: non-additive input-output association.**
(a) The $n_{pairs} = 4$ pairs of input-output associations to be learned by the network. Input patterns: flashes locate the input neurons (out of $N_{in} = 15$) subject to a strong stimulus ($f_i = f_{max}$), while the other neurons (empty squares) receive no input ($f_i = 0$). Output patterns: target activities of the $N_{out} = 15$ output neurons (black: $r_i > 0.04 r_{max}$, light gray: $r_i < 0.01 r_{max}$).
(b) Sketch of the network, with the bulk processing area including $N_{proc} = 70$ neurons, connecting the input and output neurons. The overall fractions of excitatory and inhibitory neurons in the network are equal to, respectively, 80% and 20%.
(c) Control stimulations $f_i(t)$ (see color bar for values) applied to the neurons as a function of the learning step. From top to bottom: input, output, bulk inhibitory, and bulk excitatory neurons.
(d) Costs associated with the task (green, left scale) and with regularization (orange, right scale) as functions of the learning cycle.
(e) Activities of the output neurons in response to the input stimulation patterns during learning. The color (black or light gray) of each one of the $n_{pairs} \times N_{out} = 60$ curves indicates the level of activity requested for the corresponding output neuron (high or low, see panel (a)). Training is successful when the black and light gray curves become well separated.
(f) Average activities of the output neurons for the four time steps identified with dashed line in panel (d), showing how the figures of digit emerge from the initial random image. Black and light gray activities are consistent with panel (e), with intermediate gray levels defined in the bar.

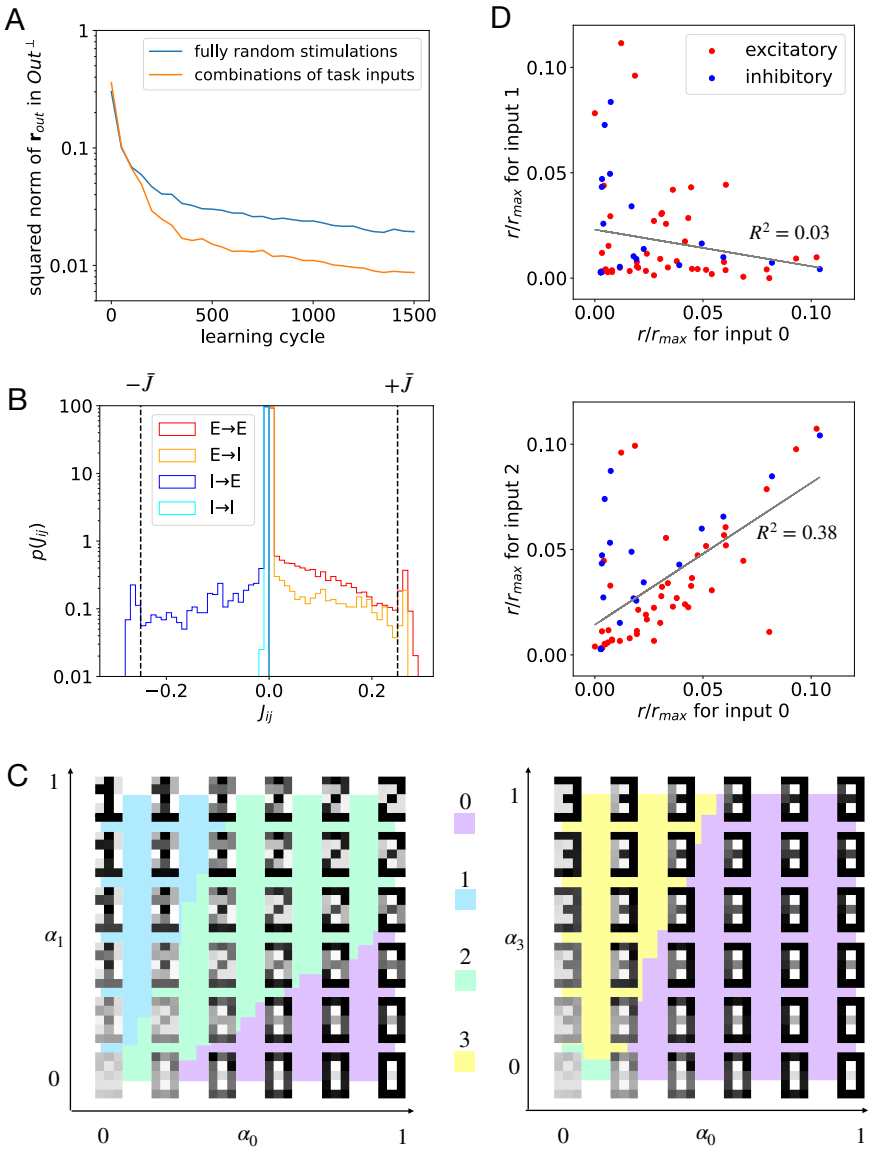

FIG. 5. **Learning dynamics and internal representations for Task 1.**
(a) Squared norm of the projection of the output activity (over $N_{out} = 15$ neurons) orthogonal to the space spanned by the $n_{pairs} = 4$ digit patterns as a function of the number of training cycles. The total activity is normalized to unity. Blue curve: case of random 0-1 stimulations over the $N_{in} = 15$ input neurons; orange: stimulations are random convex combinations of the n_{pairs} inputs. Results were averaged over 10 training trials, and each point is averaged over 50 random input stimulations.
(b) Histograms of interactions J_{ij} between the neurons in the E and I populations. Dashed lines show the soft bounds in $\pm\bar{J}$.
(c) Output neuron activities (gray levels) in response to combinations of inputs $\alpha_\mu \mathbf{f}_\mu + \alpha_\nu \mathbf{f}_\nu$, with $0 \leq \alpha_\mu, \alpha_\nu \leq 1$. The bottom left corner corresponds to vanishing input and the bottom, right and top,left corners to pure inputs associated to, respectively, digits μ and ν. The background colors (middle squares) indicate the most resembling digits. (Left) case $\mu = 0, \nu = 1$: the output of the linear combination of inputs is not the linear combination of the outputs, as required by the association task. (Right) Case $\mu = 0, \nu = 3$: the response to linear combinations of $\mathbf{f}_0$ and $\mathbf{f}_3$ is approximately additive.
(d) Scatter plot of the activities of the 70 bulk neurons for inputs $0, 1$ (top) and $0, 2$ (bottom) after training. The representations associated to patterns 0 and 1 are not correlated, as expected from the orthogonality of $\mathbf{f}_0$ and $\mathbf{f}_1$ (Fig. 4(a)). Conversely, the representations of 0 and 2 are strongly correlated, reflecting the relation $\mathbf{f}_2 = \mathbf{f}_0 + \mathbf{f}_1$ (Fig. 4(a)).

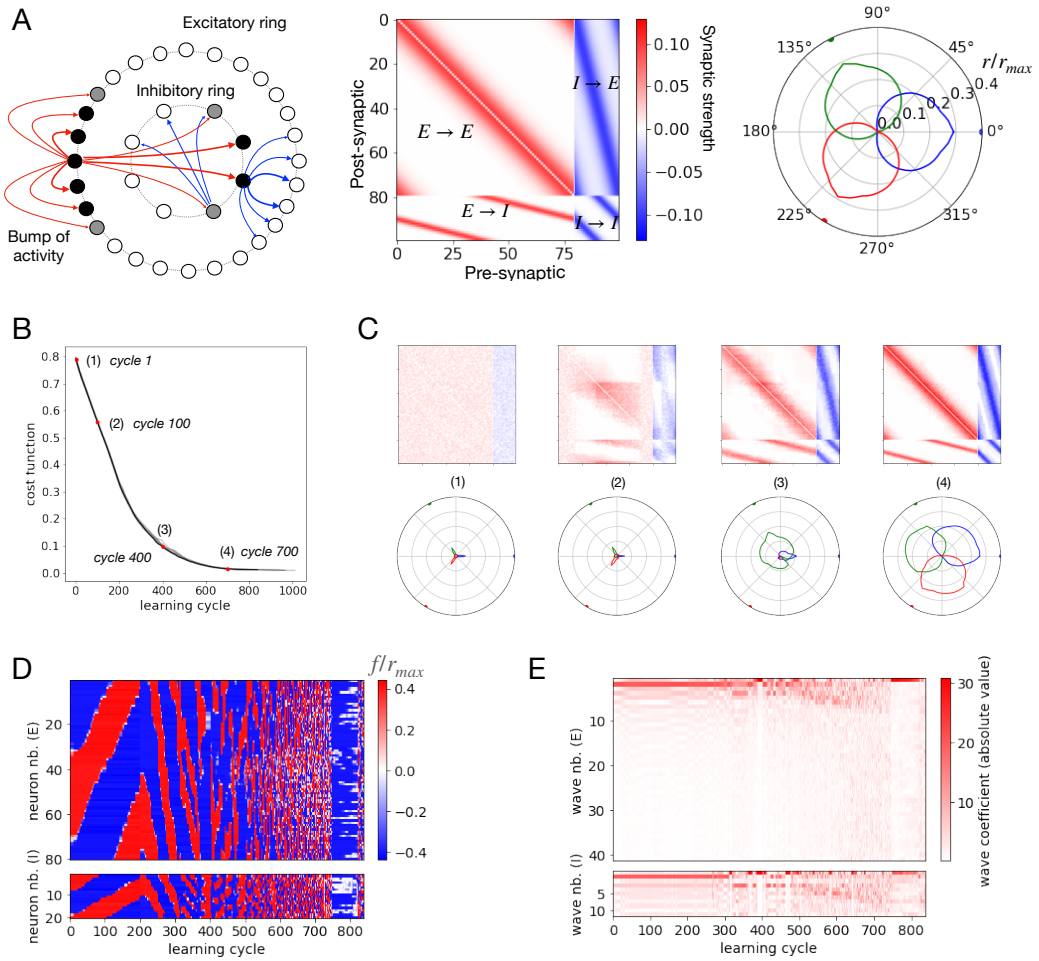

FIG. 6. **Task 2: building a continuous attractor.**
(a) Target network (left) and connectivity matrix $\mathbf{J}^{target}$ (middle). Excitatory neurons (E) are arranged on the outer ring ($N_E = 80$), and inhibitory neurons (I) on the inner one ($N_I = 20$). Neurons on the E ring have strong excitatory connections with their neighbours, and project to inhibitory neurons diametrically opposed on the I ring. Inhibitory neurons repress neighbouring neurons on the E ring and repress the opposite side of the I ring, leading to a localization of the activity (bump). Right: Radial plots of receptive fields of neurons associated to angles 0 (blue), 120 (green) and 240 (red) degrees on the E ring. A weak localized input stimulation is applied for each angle (Methods) and the activities (averaged over 100 trainings) of all neurons in the stationary state are shown.
(b) Task cost evolution during learning for 10 random naive networks (gray curves); the back line shows the average cost. Four representative learning cycles, labelled (1), (2), (3), (4) are referred to in panel (c).
(c) Connectivity matrices (top) and receptive fields (bottom) at cycles (1), (2), (3), (4), showing different stages of learning. Same color codes as in panel (a).
(d) Control stimulations $f_i(t)$ as functions of the learning cycle.
(e) Amplitude $\hat{f}^{\epsilon}_k(t) = \sum_{\ell=1}^{N_\epsilon} f_\ell(t) \cos(2\pi k\, \ell/N_\epsilon)$ of the Fourier modes $k = 0, 1, ..., N_\epsilon/2$ associated to the control stimulations shown in panel (d), for excitatory ($\epsilon = E$) and inhibitory ($\epsilon = I$) neurons. Initially, the control stimulation mainly consists of large waves (low-k Fourier modes), while at the end of training, modes at large k are used to refine the synaptic structure on short angular scales. Notice the global suppression at cycle $\simeq 750$, after the receptive fields are formed.

ACKNOWLEDGEMENTS

We thank S. Wolf for many useful discussions, and D. Chatenay for reading the manuscript. This research was supported by the European Union's Horizon 2020 research and innovation program under Grant No. 964977 (project NEU-ChiP).

DECLARATION OF INTERESTS

The authors declare no competing interests.

AUTHOR CONTRIBUTIONS STATEMENT

All authors planned and conceived the study. F.B. designed the algorithms, ran the simulations, and analyzed the results. F.B. and R.M. wrote the manuscript; All authors reviewed the manuscript.