Learning arbitrary dynamics in efficient, balanced spiking networks using local plasticity rules

Aireza Alemi[†]
Group for Neural Theory, ENS
29 rue d'ulm, Paris–75005, France
[†]alireza.alemi@ens.fr

Christian Machens
Champalimaud centre for the unknown
Avenida Brasilia, 1400-038 Lisbon, Portugal

Sophie Denève[*]
Group for Neural Theory, ENS
29 rue d'ulm, Paris–75005, France

Jean-Jacques Slotine[*]
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Abstract

Understanding how recurrent neural circuits can learn to implement dynamical systems is a fundamental challenge in neuroscience. The credit assignment problem, i.e. determining the local contribution of each synapse to the network's global output error, is a major obstacle in deriving biologically plausible local learning rules. Moreover, spiking recurrent networks implementing such tasks should not be hugely costly in terms of number of neurons and spikes, as they often are when adapted from rate models. Finally, these networks should be robust to noise and neural deaths in order to sustain these representations in the face of such naturally occurring perturbation. We approach this problem by fusing the theory of efficient, balanced spiking networks (EBN) with nonlinear adaptive control theory. Local learning rules are ensured by feeding back into the network its own error, resulting in a synaptic plasticity rule depending solely on presynaptic inputs and post-synaptic feedback. The efficiency and robustness of the network are guaranteed by maintaining a tight excitatory/inhibitory balance, ensuring that each spike represents a local projection of the global output error and minimizes a loss function. The resulting networks can learn to implement complex dynamics with very small numbers of neurons and spikes, exhibit the same spike train variability as observed experimentally, and are extremely robust to noise and neuronal loss.

1 Introduction

Recurrent networks in the nervous system perform a variety of tasks that could be formalized as dynamical systems. In many cases, these dynamical systems are learned based on examples ("desired" trajectories), a form of supervised learning. For example, to learn to control an arm, sensory-motor circuits can learn to predict both the arm state trajectories and the sensory feedbacks, that are caused by specific motor commands.

Such learning occurs under several constraints. First, synapses have only access to local information. Because any local change in a synapse could have unpredictable effects on the rest of the network,

[*]Co-PI, alphabetic order

previous approaches have often used non-local, biologically implausible learning rules such as temporal backpropagation [22] or FORCE learning [19].

Second, information in the nervous system is communicated with spikes. In order to obey the constraints imposed by the brain that is extremely costly to our metabolism [14], spiking neural networks should work with reasonably small number of spikes per dimension of the internal state dynamics. Such efficiency requires that there is no redundancy in the representation, so that spike trains of different neurons are uncorrelated.

Third, learning has to be able to resist various perturbations, such as varying levels of noise or the loss of neurons. Such robustness requires some level of degeneracy in neural populations, so that a given network has more neurons than strictly needed to learn a task.

No previous approach has been able to meet all three constraints at the same time. A few recent approaches based on adaptive control theory [18, 17] have been able to work with local learning rules, but used very cost-inefficient spiking architectures [10, 13]. Efficiency in spiking and robustness were introduced in efficient balanced networks (EBN) [3, 9], but supervised learning in these networks has so far been limited to non-local rules [15] or to linear dynamical systems [5].

In this study, we fuse the EBN framework [3, 9] with adaptive nonlinear control theory [18, 17, 16] in order to derive local learning rules for arbitrary, nonlinear dynamics, while resulting in highly efficient and robust spiking networks. The resulting learning rule inherits from adaptive control theories and EBN theories their proofs of convergence and stability, and are guaranteed to converge to the right dynamics.

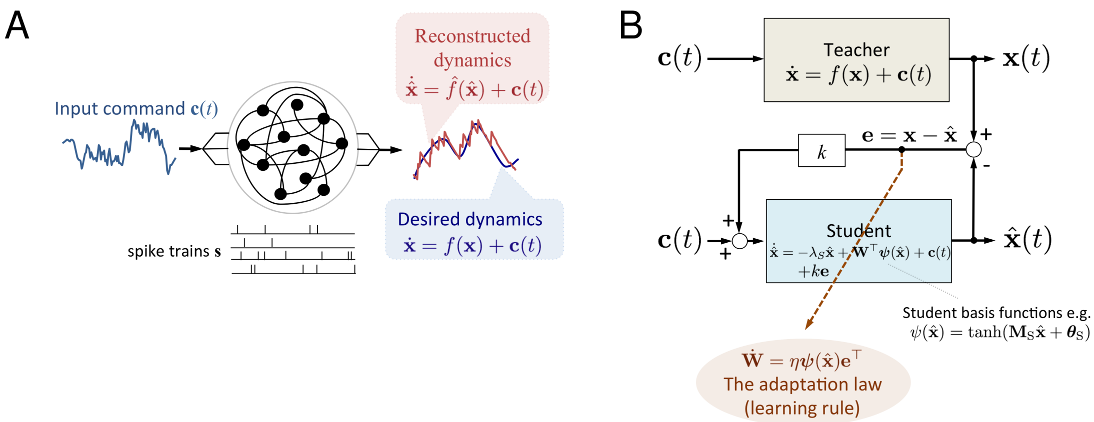

Figure 1: Schematics of the learning task and the control-theoretic solution. A. The task of learning an arbitrary dynamical system. The network is presented with a random input command with certain statistics (cyan) and it needs to produce a trajectory (red) close to the trajectory given by the teacher (blue). The supervisory error signal, i.e. $\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}}$, should be used to train the connections of the recurrent network to perform this task. B. Adaptive nonlinear control theory for solving an estimation problem formulated as a teacher-student scenario.

2 Adaptive nonlinear control theory: a teacher-student scenario

The learning task that we consider is presented in Fig. 1A. An input signal drives a recurrent spiking network. The task is to learn the network connectivity such that a reconstruction (in red with the state variable $\hat{\mathbf{x}}$) of a desired dynamics (in blue with the state $\mathbf{x}$) can be decoded from network spike trains. The learning in this task is supervised as the tracking error, $\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}}$, is available to the network. In particular, this task could correspond to a sensorimotor learning task, where the network learns a *forward internal model* that receives the *efference copy* (i.e. a copy of the input motor command) to predict the body position given the sensory error feedback and the motor command [23].

More formally, we consider the teacher dynamical system of the general form [2]

$$\dot{\mathbf{x}} = -\mathbf{x} + f(\mathbf{x}) + \mathbf{c}(t) \tag{1}$$

in which $\mathbf{x}$ is a time-dependent dynamic vector of continuous, real-valued, variables, $\mathbf{c}(t)$ is a time-varying input or command signal chosen as a filtered random signal unless otherwise stated, and $f(.)$ is an arbitrary function. Note that if the teacher dynamical system is of higher order, we can transform it into a higher-dimensional, first-order dynamical equations.

In order to understand our approach for learning, we present the estimation problem in a control-theoretic framework, then we fuse it with the EBN theory. Particularly, we use tools and results from nonlinear adaptive control theory [18, 17, 16].

The estimation problem is the following: the parameters of a "student" system need to be estimated such that its dynamics matches the dynamics of a "teacher" system (See Fig. 1B).

The student system has the following form:

$$\dot{\widehat{\mathbf{x}}} = -\widehat{\mathbf{x}} + \mathbf{W}^\top \boldsymbol{\psi}(\widehat{\mathbf{x}}) + \mathbf{c}(t) + k\,\mathbf{e}, \tag{2}$$

where $\widehat{\mathbf{x}}$ is the dynamic vector variable of the student, $\mathbf{W}$ contains the adaptive parameters of the student dynamics which are going to be learned to decrease the tracking error $\mathbf{e} = \mathbf{x} - \widehat{\mathbf{x}}$, and $\boldsymbol{\psi}(\widehat{\mathbf{x}})$ are the student basis functions (e.g. $\boldsymbol{\psi}(\widehat{\mathbf{x}}) = \tanh(\mathbf{M}_{\mathrm{S}}^\top \widehat{\mathbf{x}} + \boldsymbol{\theta}_{\mathrm{S}})$). The feedback gain factor k causes the student to follow the desired dynamics closely which provides the opportunity for adjusting the adaptive parameters $\mathbf{W}$ in order to learn the dynamics. The higher the gain k, the closer the student trajectories are to the desired ones. In order to get a good generalization behavior and fewer training iterations, after certain number of training iterations, it is a good practice to decrease k. During the test phase, we do not provide this feedback, i.e. $k_{\text{test}} = 0$.

If the student has sufficiently many rich basis functions to approximate the function $f(.)$, then there exists a solution called $\mathbf{W}^{\text{true}}$. If we define a Lyapunov function as

$$V = \frac{1}{2}\mathbf{e}^\top\mathbf{e} + \frac{1}{2\eta}\,\mathrm{Tr}\left(\widetilde{\mathbf{W}}^\top\widetilde{\mathbf{W}}\right), \tag{3}$$

where $\widetilde{\mathbf{W}} = \mathbf{W}^{\text{true}} - \mathbf{W}$ is the estimation error and $\mathrm{Tr}(.)$ is the matrix trace operator, it can be shown (see Supplementary Materials) that the following adaptation law (or the learning rule)

$$\dot{\mathbf{W}} = \eta\,\boldsymbol{\psi}(\widehat{\mathbf{x}})\mathbf{e}^\top, \tag{4}$$

where η is a learning rate, will decrease the Lyapunov function, i.e. $\dot{V} = -(k+1)\,\mathbf{e}^\top\mathbf{e} \leq 0$. This result together with boundedness conditions for V and $\ddot{V}$ guarantees that $\dot{V}$, and hence the tracking error $\mathbf{e}$, will asymptotically go to zero and that the system is asymptotically stable [17] (See Supplementary Materials). Moreover, if the input $\mathbf{c}(t)$ is rich enough, then $\mathbf{W} \to \mathbf{W}^{\text{true}}$.

3 Learning a functional spiking network

We want to translate our student into a recurrent network of N leaky integrate-and-fire (LIF) neurons. Previous work has shown how to implement arbitrary linear or nonlinear dynamical systems in efficient balanced networks [3, 20]. EBN theory is based on two assumptions. First, an estimate of the K-dimensional state variable, $\mathbf{x}$, can be extracted from the filtered spike trains, $\mathbf{r}$, using a linear decoder, such that

$$\widehat{\mathbf{x}} = \mathbf{D}\mathbf{r}, \tag{5}$$

where $\mathbf{D}$ is a fixed decoding weight matrix of size $K \times N$. Second, neurons in the network fire spikes such that this estimate, $\widehat{\mathbf{x}}$, closely follows the true state variable, $\mathbf{x}$, under cost constraints. Specifically, the network minimizes the following objective function,

$$\mathcal{L} = \langle L \rangle = \left\langle \|\mathbf{x} - \widehat{\mathbf{x}}\|^2 + \mu\|\mathbf{r}\|_2^2 + \nu\|\mathbf{r}\|_1 \right\rangle. \tag{6}$$

[2]Our dynamic variables $\mathbf{x}(t)$ and input signals $\mathbf{c}(t)$ are a function of time but sometimes for simplicity we omit the time variable t, writing them as $\mathbf{x}$ and $\mathbf{c}$

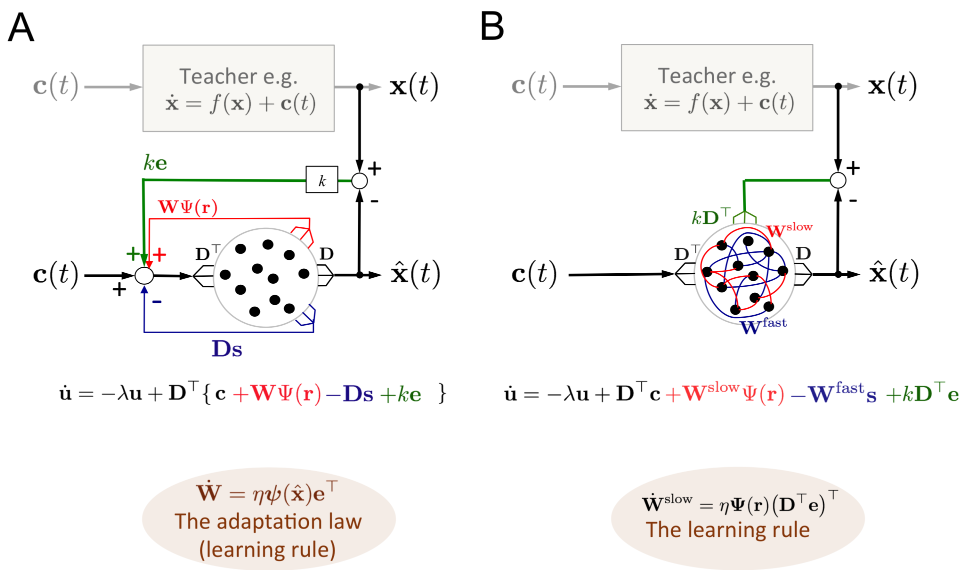

Figure 2: Building a spiking network approximating arbitrary dynamical systems. A. The unfolded version of the network of LIF neurons (with membrane potentials $\mathbf{u}$ and a threshold and reset mechanism) with three feedback loops implementing desired dynamics of $\mathbf{x}$ in the network: the role of red loop is to implement nonlinear dynamical system, the blue one provides efficiency and robustness, and the green one feeds the error back into the network such that $\widehat{\mathbf{x}}$ closely tracks $\mathbf{x}$. B. The folded version of the network with slow and fast connections. The corresponding learning rules are shown at the bottom.

where $\|\cdot\|_p$ denotes the Lp-norm, $\langle\cdot\rangle$ is an average over time, and μ and ν determine the costs associated with spiking. The first term on the right-hand side of Eq. 6 ensures a good reconstruction, whereas the L2 and L1 cost functions on neural activity ensure a distributed and sparse spiking activity, respectively [3].

The resulting network consists of LIF neurons whose membrane potentials obey the equation [3, 4]

$$\dot{\mathbf{u}} = -\mathbf{u} + \mathbf{D}^\top(\mathbf{x} + \dot{\mathbf{x}}) - \mathbf{W}^{\text{fast}}\mathbf{s}, \tag{7}$$

and whose firing thresholds are given by $T_i = \frac{1}{2}(\|\mathbf{D}_i\| + \mu + \nu)$, where $\mathbf{D}_i$ is the i-th column of $\mathbf{D}$. The thresholds ensure that each neuron fires only when its spike decreases the objective function (Eq. 6), and the "fast" recurrent connections, $\mathbf{W}^{\text{fast}} = \mathbf{D}^\top\mathbf{D} + \mu\mathbf{I}$, where $\mathbf{I}$ is the identity matrix, make sure that the membrane potentials properly track the objective function. Indeed, the voltage of each neuron is simply a projection of the reconstruction error so that $\mathbf{u} = \mathbf{D}^\top(\mathbf{x} - \widehat{\mathbf{x}}) - \mu\mathbf{r}$.

In order to implement arbitrary dynamical systems in this framework, given by $\dot{\mathbf{x}} = -\mathbf{x} + f(\mathbf{x}) + \mathbf{c}$, we simply replace the term $\dot{\mathbf{x}} + \mathbf{x}$ on the right-hand-side of Eq. 7, to obtain

$$\dot{\mathbf{u}} = -\mathbf{u} + \mathbf{D}^\top\mathbf{c} - \mathbf{W}^{\text{fast}}\mathbf{s} + \mathbf{D}^\top f(\mathbf{x}). \tag{8}$$

We can now include two approximations. First, we can approximate f as a sum of the "student" basis functions (similar to Eq. 2), so that $f(\mathbf{x}) = \sum_i \mathbf{W}_i\phi_i(\mathbf{x})$, where $\mathbf{W}_i$'s are the columns of $\mathbf{W}$. Second, we can self-consistently replace the state $\mathbf{x}$ by its estimate based on network activity (Eq. 5). In turn, we obtain (Fig. 2B)

$$\dot{\mathbf{u}} = -\mathbf{u} + \mathbf{D}^\top\mathbf{c} - \mathbf{W}^{\text{fast}}\mathbf{s} + \mathbf{D}^\top\mathbf{W}^\top\phi(\mathbf{Dr}). \tag{9}$$

From a biological point-of-view, the last term cooresponds to nonlinear and highly structured dendrites. While dendrites are often nonlinear, the required detailed spatial organization of their inputs is rather speculative. To relax this latter constraint, we can take advantage of the fact that the network's internal state dynamics is extremely robust to large amounts of noise (as ensured by its constant, greedy minimization of its own coding errors), as long as $N > 2K$. Thus, we can replace $\boldsymbol{\phi}(\mathbf{Dr})$ with an unstructured, nonlinear dendrite that receives random connections from other neurons, $\Psi_i(\mathbf{r}) = \boldsymbol{\phi}(\mathbf{M}_i^\top \mathbf{r} + \boldsymbol{\theta}_i)$ where $\mathbf{M}_i$'s are the columns of the matrix $\mathbf{M}$. Indeed, the quantity $\mathbf{M}^\top \mathbf{r}$ can be written as the sum of its projection onto the decoder $\mathbf{D}$, and its projection onto the null space of $\mathbf{D}$ (denoted by $\mathbf{D}^\emptyset$), so that $\mathbf{M}^\top \mathbf{r} = \mathcal{M}^\top \hat{\mathbf{x}} + \mathcal{M}^\top \mathbf{D}^\emptyset \mathbf{r}$. Given that EBNs do not control spiking in the null space of the decoder, the second term acts as unstructured noise. [3]

To clarify the learning problem, we will write the resulting student network dynamics in the general form

$$\dot{\mathbf{u}} = -\mathbf{u} + \mathbf{Fc} - \mathbf{W}^{\text{fast}}\mathbf{s} + \mathbf{W}^{\text{slow}}\boldsymbol{\Psi}(\mathbf{r}). \tag{10}$$

We note that the network has three sets of synaptic connections. The feedforward connections, $\mathbf{F}$, receive and weight the external signal input, $\mathbf{c}$. The fast connections, $\mathbf{W}^{\text{fast}}$, guarantee the proper and efficient distribution of spikes across the network. Finally, the slow connections, $\mathbf{W}^{\text{slow}}$, implement the dynamics of the student system. In previous work, we have shown how to learn the feedforward connections [6] and the fast recurrent connections [6, 4]. Though all connections could be trained simultaneously, we here concentrate on how to train the slow connections based on example trajectories from an unknown (teacher) dynamical system. We use a fixed random decoding weight matrix $\mathbf{D}$ (which is close to optimal in the case of uncorrelated command signals) and set the feedforward connections and recurrent connections to their optimal values, $\mathbf{F} = \mathbf{D}^\top$ and $\mathbf{W}^{\text{fast}} = \mathbf{D}^\top\mathbf{D} + \mu\mathbf{I}$.

A direct translation of the adaptive control to the neural network will finally permit us to define a local learning rule for the slow connections. Rather than feeding back the error into the student network by adding it to its input, we directly inject the errors as feedback to each neuron with connections $\mathbf{D}^\top$, which is mathematically equivalent. In the presence of this feedback control loop, the network equation becomes

$$\dot{\mathbf{u}} = -\mathbf{u} + \mathbf{Fc} - \mathbf{W}^{\text{fast}}\mathbf{s} + \mathbf{W}^{\text{slow}}\boldsymbol{\Psi}(\mathbf{r}) + k\mathbf{D}^\top\mathbf{e}. \tag{11}$$

Moreover $\mathbf{W}^{\text{slow}} = \mathbf{D}^\top\mathbf{W}^\top$ is directly related to the coefficients $\mathbf{W}$ of the basis functions in the control theory framework. This allows us to map the adaptation law (Eq. 4) for the coefficients of the basis function to the following learning rule for the slow connections. Replacing $\mathbf{e}$ in Eq. 4 with $\mathbf{D}^\top\mathbf{e}$, we obtain

$$\dot{\mathbf{W}}^{\text{slow}} = \eta\, \boldsymbol{\Psi}(\mathbf{r})(\mathbf{D}^\top\mathbf{e})^\top. \tag{12}$$

In other words, the resulting learning rule is the product of a pre-synaptic input (passed through the dendritic nonlinearity) and the error feedback received by the post-synaptic neuron. Hence, the learning is local.

For the case of sensorimotor learning, the quantity $k\mathbf{e}$ could correspond to visual or somatosensory "prediction errors", e.g. the difference between the sensory input and its prediction based on the efferent copy of the motor commands. Over the course of learning, the errors become smaller and would eventually vanish in the absence of motor noise or sensory noise. Consequently, the feedback would become silent and could be removed entirely.

The general idea of this derivation, and the mapping from the control-theoretical framework onto efficient balanced networks, is also illustrated in Fig. 2A.

4 Implementing nonlinear dynamics

We employed the framework described above in the following example tasks.

[3] We note that it would be possible to train the parameters $\mathbf{M}$ using unsupervised learning, since $\mathbf{D}^\top$ corresponds to the eigenvectors with non-zero eigenvalues in the neural correlation matrix. This should render the network even more efficient and robust as will be explored in the future.

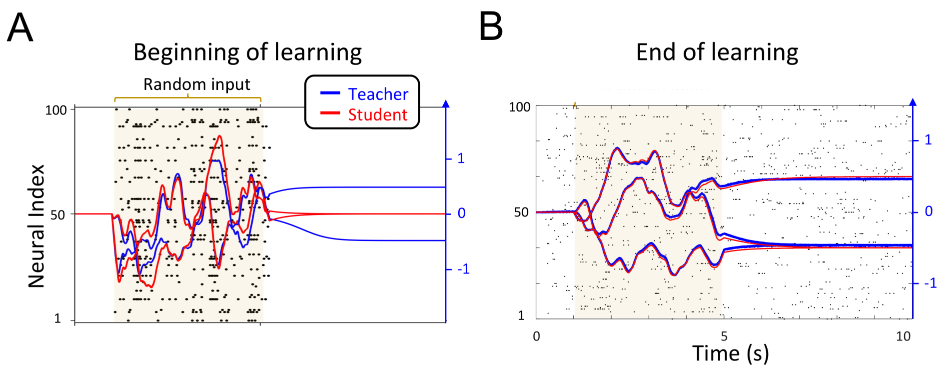

Figure 3: Result of learning a bistable attractor in an efficient, balanced spiking network. The desired dynamics given by a teacher is $\dot{x} = x(0.5 - x)(0.5 + x) + c(t)$ where $c(t)$ is a random input signal. A .In the beginning of learning, the reconstructed dynamics of the student network (in red) during the presentation of the input (shaded area) is due to the random inputs. After the input is turned off, the reconstruction does not go to any of the attractor states. B. At the end of learning, when the input is off, the reconstructed signal falls into one of the stable attractors.

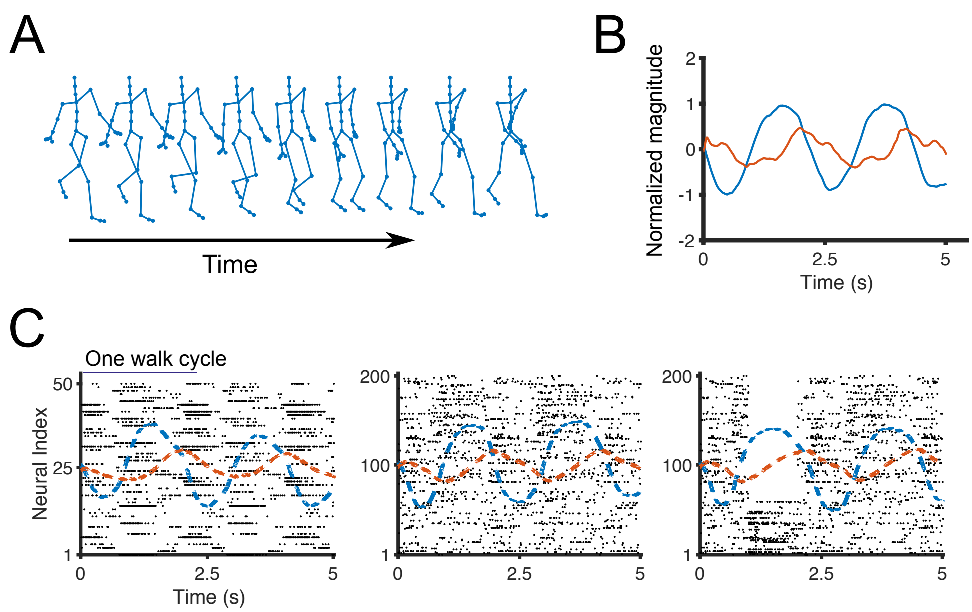

Figure 4: Result of learning to walk in spiking networks. A. The dynamics of the walk of a sticky man implemented by a network of spiking model neurons. The training data is from CMU Motion Caption Library. B. Two first principal component of the desired trajectory of the walk. The original dataset for the walk has 62 channels. C. Raster plots for networks that learned a walking dynamics based on four principal components of a trajectory of the walk. The left panel shows the plot for a network of 50 neurons. Four principal components were provided to the network as desired trajectories. The reconstruction of the first two principal components are shown overlaid on the raster plot. As the number of neurons in the network increases to 200 neurons (middle panel) the firing rates decrease and spiking activity becomes more asynchronous and irregular and the reconstruction improves. The network is drastically robust to silencing neurons as shown in the right-most panel where 70% of the neurons are silenced for a period of around one second but the network continues to generate walking with a short, negligible distortion. Note that after silencing the neurons, the network weights do not undergo any learning, but other active neurons compensate for neuronal loss in the network.

4.1 Bistable attractor

The bistable attractor is an important nonlinear dynamical system which is widely studied in systems neuroscience. We implemented the following one-dimensional attractor dynamics

$$\dot{x} = x(0.5 - x)(0.5 + x) + c(t), \tag{13}$$

where $c(t)$ is a random input command. This system has two stable fixed-point attractors at $x = \pm 0.5$ and a saddle-point at $x = 0$. We used $N = 50$ neurons with random readout weights $\mathbf{D}$ to implement this system. As shown in Fig. 3A, the random input $c(t)$ drives the student network in the beginning of learning. Once this input is zero the desired dynamics given by the teacher (in blue) goes to one of the stable attractors. But the decoded dynamical variable of the network (in red) has not yet learned the desired attractors. After learning (Fig 3B), the network closely follows the desired trajectories in response to a random unseen input (not used during training) and, more importantly, once the input is turned off, it settle on one of the attractors. Note that the network exhibits realistic spiking activity with low firing rates and asynchronous, irregular spiking activity. The overall number of spikes is drastically lower than that of spiking networks that effectively use *rate codes* [12, 11, 13].

4.2 Motion capture example

We also implemented the walking dynamics from the database of Carnegie Mellon University Motion Capture Library (MOCAP) `http://mocap.cs.cmu.edu/`. We used a similar procedure as in [19]. Briefly, the data was taken from file "08_01.amc" which consisted of 62 channels out of which three were set to zero, converting the movement to a walk on a treadmill. We preprocessed the data by a moving average.

In order to reduce the dimensions of the input signal, we took the first four principle components. This reduced input closely follows the original walking dynamics with minor loss. As the system is a mechanical system, and therefore of second order, we require both velocity and position information. Therefore, in order to be able to model it, we fed both the trajectories and its derivatives to the network, so that the overall input was 8-dimensional. Similarly, we provided a linear combination of the position error and the velocity error as the error feedback to the network. We then learned the dynamics of the 8-dimensional input with two different networks, one of size $N = 50$ and and one of size $N = 200$ (compare with [19]).

Fig. 4A shows the learned walking, and the left and middle panels of Fig. 4B shows the raster plot of the 50-neuron and 200-neuron networks, respectively. An important feature of our model is its expansion: the number of input channels (K) must be smaller than the number of neurons (N) in the network. In order for the network to have the desired aforementioned properties, as a rule of thumb, the expansion ratio ($\Lambda = N/K$) should be larger than 5–10. This expansion provides many possible solutions for the network to perform a task. Thanks to the efficiency principle, the network is able to choose the most efficient one. It should be noted that the realistic spiking activity is closely linked to the expansion: increasing the expansion ratio results in lower firing rate and more irregular spiking activity (compare left and right panels of Fig. 4C). This is one of the main differences with other spiking networks (but in fact they use a *rate code*) implementing complex dynamics such as the Neural Engineering Framework (NEF) [12, 11], as they would typically need firing rates in the order of the inverse of synaptic time-scales to learn properly. Another striking feature of the model is its robustness to neural death and noise. Thanks to the presence of the fast connections, even if $\%70$ of the neurons in the network are silenced the network would still be able to perform the task with minimal loss in the performance (right-most panel in Fig. 4C). This is consistent with the expected robustness of the EBN framework [2].

5 Discussion

We have proposed a local learning rule in an spiking neural network of LIF neurons for learning arbitrary complex dynamics. The resulting networks exhibit low firing rates with asynchronous, irregular spiking activity where the spiking representation is as efficient as possible. The efficiency principle that we have exploited has direct consequences: the inhibitory input currents in each neuron closely track the excitatory input (E/I balance); and the network is highly robust to noise, neural elimination, and uncertainty in desired dynamics [3, 2]. The learning rule is obtained thanks to concepts and tools in nonlinear adaptive control theory. This approach has the benefit of providing a systematic way to study convergence and stability properties of the learning process which currently is not pursued in the mainstream learning procedures in neuroscience for spiking neurons [19, 1, 20]. Studying the effect of synaptic delays needs to be addressed in future versions of our model, although previous work suggests that increasing the amount of noise may help avoiding oscillation and synchronization [7].

The different parts of the spiking network have straightforward biological interpretations. Fast connections could correspond to interneurons which would mostly be driven by monosynaptic, fast AMPA synapses from excitatory neurons, targeting the soma and relying on ionic (GABA-A) neurotransmission. Slow connection could be implemented by slower metabolic channels (e.g. NMDA/GABA-B) and correspond either to direct connections between pyramidal cells, or disynaptic inhibition using another type of interneurons, in the case of negative weights.

The network seems to be highly structured in our framework. However, it has already been shown that all the other connections in the network can be trained using unsupervised, local spike-time-dependent plasticity rules with the exception of $\mathbf{W}^{\text{slow}}$ which is the contribution of the present work. In particular, the fast connections $\mathbf{W}^{\text{fast}}$ can be trained using local plasticity rules, while the feedforward connections (and thus, the decoding weights) can be trained using a Hebbian spike-time-dependent rule [6, 4] . Note that these inhibitory fast connections could be thought of as implementing a kind of 'dynamic' Winner-Take-All (WTA) network where once a neuron wins and spikes, then other neurons have a chance to fire as well. This is a different scheme from a distributed WTA of inhibitory neurons [21].

In our general framework, nonlinear dendrites compute the basis functions. However, we have a large degree of freedom in the choice of $\mathbf{M}$ and the nonlinearities, i.e., in the design of the dendritic tree. The case that might be easiest to map to the control-theoretic framework is the case where $\mathbf{M} = \mathbf{D}\mathbf{M}'$ is a low rank matrix. But other cases such as random $\mathbf{M}$ also work in practice. We provided intuitions for why such a suboptimal architecture still functions: the network activity is forced to be as efficient as possible due to the presence of the fast connections (E/I balance). This defines an optimal combination of neural activities for each state $\widehat{\mathbf{x}}$. In other words, neural firing rates become almost completely determined by the internal state estimate (up to their "poisson-like" variability). This neural activity projected onto an arbitrary matrix will also be a function of $\hat{\mathbf{x}}$, plus some unstructured input that may be considered as noise. This noise is automatically compensated by the network robustness. In the case of the walking example, we tested a diagonal matrix $\mathbf{M}$ (in which case, the dendritic nonlinearity is replaced by a nonlinear transfer function for each neuron) and the network was still able to learn without any difficulty. The best choice of $\mathbf{M}$ (including the possibility of learning these parameters) remains to be explored further elsewhere. Our approach is aligned with other works taking similar strategies for implementing nonlinearities in recurrent networks [1, 20].

Using nonlinear adaptive control theory, one can come up with a local learning rule for the Neural Engineering Framework to learn complex dynamics [13], however these networks don't have the fast connections to provide efficiency and balance. More importantly, they do not scale up easily as they would need overall lots of spikes to implement dynamics [19, 13]. They are not able to exploit the expansion completely, therefore it would be more challenging for these networks to exhibit realistic spiking activity and robustness. Our work is different, as it seamlessly blends and exploits earlier work on EBNs, which exploited the spiking nature of neural activity (rather than treating it as a hindrance).

As a conclusion, we argue for a close relationship between network efficiency and robustness and the tight E-I balance observed in cortical circuits. We suggest that experimentally observed spike trains, with low-firing rate and asynchronous, irregular spike trains, are a signature of an efficient spike-based coding, and not a noisy rate-based population code. The network effectively implements dimensionality reduction: regardless of its size, the dimensionality of its population dynamics and recurrent weights eventually becomes restricted to the dimensionality of the task, while neural fluctuations occur in direction orthogonal to the task. Thus, we predict that such low-dimensional dynamics emerge through experience in biological neural circuits. Finally, learning in biological circuits would require that feedback connections monitoring the network performance both drive the neurons and modulate learning. Each slow connection is learned as a function of the correlation between pre-synaptic input rate and postsynaptic error feedback, until this error feedback is canceled. Thus, only neurons with correlations to the error (and presumably contributing to such error) see their synaptic weights change. In contrast, backpropagation would result in diffuse change in the entire network. These predictions have broad implications for Brain Machine Interfaces. In the near future, this theory may pave the way for implementing more complex tasks and for spike based unsupervised, hierarchical and reinforcement learning, in both biological and artificial spiking networks.

The framework presented here can have engineering applications for example in light-weight robots or robots that are sent to space missions where efficiency matters. Furthermore, we have used the framework to estimate the parameters of a desired system but it can also be used for adaptive nonlinear control applications — where the controller needs to adapt to unknown changes in dynamics and kinematics of robots [8, 10] — to give an efficient adaptive spiking controller. The current framework is obtained for deterministic dynamical systems – future work will extend it to stochastic dynamical systems.

References

[1] L. F. Abbott, B. DePasquale, and R.-M. Memmesheimer. Building functional networks of spiking model neurons. *Nature Publishing Group*, 19(3):350–355, Feb. 2016.

[2] D. Barrett, S. Deneve, and C. K. Machens. Optimal compensation for neuron loss. *eLife*, 5:e12454, 2016.

[3] M. Boerlin, C. K. Machens, and S. Deneve. Predictive Coding of Dynamical Variables in Balanced Spiking Networks. *PLoS computational biology*, 9(11):e1003258, Nov. 2013.

[4] R. Bourdoukan, D. Barrett, C. K. Machens, and S. Deneve. Learning optimal spike-based representations. *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[5] R. Bourdoukan and S. Deneve. Enforcing balance allows local supervised learning in spiking recurrent networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 982–990, 2015.

[6] W. Brendel, R. Bourdoukan, P. Vertechi, C. K. Machens, and S. Deneve. Learning to represent signals spike by spike. *arXiv.org:1703.03777*, 2017.

[7] M. Chalk, B. Gutkin, and S. Deneve. Neural oscillations as a signature of efficient coding in the presence of synaptic delays. *eLife*, 2016.

[8] C. C. Cheah, C. Liu, and J. Slotine. Adaptive tracking control for robots with unknown kinematic and dynamic properties. *The International Journal of Robotics Research*, 25(3):283–296, Mar. 2006.

[9] S. Deneve and C. K. Machens. Efficient codes and balanced networks. *Nature Neuroscience*, 2016.

[10] T. DeWolf, T. C. Stewart, J.-J. Slotine, and C. Eliasmith. A spiking neural model of adaptive arm control. 283(1843):20162134, 2016.

[11] C. Eliasmith. A unified approach to building and controlling spiking attractor networks. pages 1–39, Mar. 2005.

[12] C. Eliasmith and C. H. Anderson. *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. MIT press, 2004.

[13] A. Gilra and W. Gerstner. Predicting non-linear dynamics: a stable local learning scheme for recurrent spiking neural networks. *arXiv preprint arXiv:1702.06463*, 2017.

[14] S. B. Laughlin, R. R. de Ruyter van Steveninck, and J. C. Anderson. The metabolic cost of neural information. *Nature Neuroscience*, 1(1):36–41, May 1998.

[15] R.-M. Memmesheimer, R. Rubin, B. P. Ölveczky, and H. Sompolinsky. Learning Precisely Timed Spikes. *Neuron*, 82(4):925–938, May 2014.

[16] R. M. Sanner and J.-J. Slotine. Gaussian networks for direct adaptive control. *IEEE Transactions on neural networks*, 3(6):837–863, 1992.

[17] J. Slotine and W. Li. *Nonlinear applied control*. Prentice Hall, 1991.

[18] J. J. E. Slotine and J. A. Coetsee. Adaptive sliding controller synthesis for non-linear systems. *International Journal of Control*, 43(6):1631–1651, June 1986.

[19] D. Sussillo and L. F. Abbott. Generating Coherent Patterns of Activity from Chaotic Neural Networks. *Neuron*, 63(4):544–557, Aug. 2009.

[20] D. Thalmeier, M. Uhlmann, H. J. Kappen, and R.-M. Memmesheimer. Learning Universal Computations with Spikes. *PLoS computational biology*, 12(6):e1004895, June 2016.

[21] W. Wang and J.-J. E. Slotine. Fast computation with neural oscillators. *Neurocomputing*, 69(16-18):2320–2326, Oct. 2006.

[22] P. J. Werbos. Backpropagation through time: what it does and how to do it. In *Proceedings of the IEEE*, 1990.

[23] D. M. Wolpert and Z. Ghahramani. Computational principles of movement neuroscience. *Nature Neuroscience*, 2000.