The role of balance between excitation and inhibition in learning in spiking networks

Ralph Bourdoukan

UPMC SORBONNE UNIVERSITÉS

THÈSE DE DOCTORAT DE
l'UNIVERSITÉ PIERRE ET MARIE CURIE

Spécialité

Science du Vivant

École doctorale cerveau, cognition et comportement

Présentée par

Ralph Bourdoukan

Pour obtenir le grade de

DOCTEUR de l'UNIVERSITÉ PIERRE ET MARIE CURIE

Sujet de la thèse :

Le rôle de la balance entre excitation et inhibition dans l'apprentissage dans les réseaux de neurones à spikes

soutenue le 10 octobre 2016

devant le jury composé de :

Mme. Claudia Clopath	Rapporteur
M. Raoul-Martin Memmesheimer	Rapporteur
M. Mehdi Khamassi	Examinateur
M. Jean-Pierre Nadal	Invité
Mme. Sophie Denève	Directrice de thése

Acknowledgments

I would like to thank my advisor Sophie Denève for her supervision and guidance. It was a pleasure working with someone with such scientific and human qualities. I would also like to thank Christian Machens for his support, collaboration as well as all the interesting discussions we had.

Sincere and deep gratitude to my teacher Saab Abou Jaoude for his extreme generosity and support during my prepa.

I would like to express my heartfelt gratitude to my brother Paul, as well as my father and my mother, for their support and sacrifices during my studies. Many thanks to Eva, Henry, Mario and Christophe and especially to my sister Chrystelle, my closest companion during these years.

I would also like to thank my dear friends Patrick and Matthew for proofreading the introduction and the discussion. Most importantly, I am thankful for their presence and support, which made this year much more pleasant. I would also like to thank my friends and colleagues Dani and Gabrielle for proofreading the third article.

During all my higher studies I had the chance to be surrounded by great people. I would like to thank all my dear friends for their support.

Finally, I thank all my colleagues in the GNT for all the interesting scientific discussions and for making the working environment very pleasant.

Résumé

Pour effectuer une tâche, le cerveau doit représenter et manipuler les stimuli. Par exemple, lorsqu'on saisit un objet, le cerveau doit représenter la scène visuelle, identifier l'objet et produire des commandes motrices appropriées qui auront comme conséquence le mouvement correct du bras. Ce processus requiert la représentation de variables continues, comme l'image et la position du bras, par les circuits neuraux qui communiquent à l'aide de potentiels d'action discrets. Comment le cerveau représente ces quantités continues avec les potentiels d'action qui sont de nature discrète? Comment ces représentations sont-elles distribuées dans les populations de neurones? Depuis les enregistrements d'Adrian, on suppose communément que les neurones représentent l'information continue à l'aide de leur fréquence de décharge. En outre, beaucoup de modèles théoriques supposent que les neurones représentent l'information indépendamment les uns des autres. Cependant, un tel codage indépendant semble très inefficace puisqu'il exige la génération d'un très grand nombre de potentiels action afin d'atteindre un certain niveau de précision. Dans ces travaux, on montre que les neurones d'un réseau récurrent peuvent apprendre - à l'aide d'une règle de plasticité locale - à coordonner leurs potentiels d'actions afin de représenter l'information avec une très haute précision tout en déchargeant de façon minimale. Le nombre de potentiels d'action émis est très inférieur à celui requis par un codage indépendant. La dérivation de la règle de plasticité se fonde sur l'équivalence entre le codage efficace et l'équilibre précis entre l'excitation et l'inhibition dans chaque neurone. Ainsi, La règle n'est pas conçue afin d'optimiser directement la sortie du réseau, mais pour imposer un équilibre entre excitation et inhibition. Un tel objectif - atteindre l'équilibre entre l'excitation et l'inhibition - permet d'obtenir une règle d'apprentissage locale et biologiquement plausible. La règle d'apprentissage agit uniquement sur les connexions récurrentes dans le réseau. Si l'apprentissage des connexions récurrentes est combiné avec une règle d'apprentissage pour les connexions d'entrée du réseau, les neurones s'adapteront également aux statistiques des signaux d'entrée. En effet, la règle d'apprentissage pousse graduellement le vecteur de poids d'entrée de chaque neurone vers la direction du stimulus moyen qui cause son déchargement. Ce faisant, elle garantie que les directions et caractéristiques importantes des entrées soient représentées. Ceci augmente l'exactitude de la représentation. Finalement, nous montrons que si un réseau récurrent a des connexions latérales avec deux échelles de temps différentes - qu'on appelle rapide et lente - il peut apprendre à effectuer des transformations dynamiques et complexes sur ses entrées. Plus précisément, il apprend à produire des sorties qui ont une dynamique temporelle linéaire spécifique, la relation d'entrée-sortie du réseau étant spécifiée par une équation différentielle linéaire. Pour pouvoir apprendre, le réseau doit recevoir un signal d'erreur fort et spécifique. Les connexions rapides dans le réseau sont modifiées à l'aide de la règle qui équilibre l'excitation et inhibition décrite précédemment. Cette règle engendre une coordination serrée entre les neurones et distribue l'information concernant

la sortie globale du réseau avec précision. Une telle représentation distribuée et précise permet la dérivation d'une règle d'apprentissage locale pour les connexions lentes. Les connexions lentes sont responsables de la génération de la dynamique temporelle de la sortie du réseau. La règle d'apprentissage pour ces connections est simple, elle est proportionnelle au taux de décharge du neurone présynaptique et au signal d'erreur reçue sous la forme d'un courant par le neurone postsynaptic. Dans tous ces réseaux, le stochasticité du temps de décharge d'un neurone n'est pas une signature d'un bruit mais de la dégénérescence de la représentation. Puisque plusieurs neurones représentent la même caractéristique, beaucoup de combinaisons différentes de potentiels d'actions produisent le même signal en sortie. Par conséquence, même avec un bruit très petit dans le réseau les trains de potentiels d'action sembleront complètement aléatoires et seront non reproductible d'un essai à l'autre. Ces travaux donnent ainsi une interprétation radicalement différente de l'irrégularité trouvée dans des trains de potentiels d'actions dans le cerveau. Ici, le caractère aléatoire n'est pas une signature d'un bruit mais au contraire de précision et d'efficacité. En raison de leur encodage efficace, ces réseaux n'ont pas besoin de beaucoup de neurones pour réaliser une certaine fonction par opposition à d'autres approches comme le "Liquid Computing".

Abstract

To perform a task, the brain has to represent stimuli and act on them. For example, while reaching for an object, the brain has to represent the visual scene, recognize the object and generate the appropriate motor commands that will result in the correct arm movement. Performing this task requires the representation of continuous quantities by the neural circuits which use discrete action potentials to communicate. Indeed, the image and the arm position all evolve in a continuous space. The first question that comes up is how the brain represents these continuous quantities with discrete spikes and how these representations are shared among neurons? Since the recordings by Adrian, neurons were thought to represent quantities through their firing rates and individual spikes being considered as noisy. In addition, many theoretical models assume that neurons represent information independently from one another. However, such an independent rate coding seems very inefficient since it requires a very large number of spikes to achieve a certain level of precision. In this work, we show that leaky integrate-and-fire neurons in a recurrent network can learn, using a plasticity rule, to coordinate their spikes and represent information very precisely while firing a small amount of spikes. The number of spikes fired is much lower than what an independent rate code requires. The derivation of the plasticity rule relies on the equivalence between efficient coding and a tight balance between excitation and inhibition in each neuron. The learning rule is thus not designed to directly optimize the network's output, but to enforce precise balance on the level of a single neuron. Such an objective - reaching the balance between excitation and inhibition - results in a local biologically plausible learning rule. The learning rule acts on the recurrent connections in the network. If the learning of the recurrent connections is combined with a learning rule for the feedforward connections, the neurons will also adapt to the input statistics. The feedforward learning rule gradually pushes the feedforward weight vector of each neuron towards the direction of its spike triggered average. By doing so, it ensures that all the relevant directions and features of the inputs are represented. Indeed, this increases the accuracy of the representation. Finally, we show that if a recurrent network has lateral connections with two different time scales, fast and slow, it can learn to operate interesting transformations on its inputs. More precisely, it learns to generate outputs that have specific linear temporal dynamics, the input/output relationship being specified by a linear differential equation. To be able to learn, the network must receive a strong and specific error feedback. The fast connections in the network undergo the same balancing learning rule described previously. This rule enforces a tight coordination between neurons and precisely distributes the information concerning the output of the network across neurons. Such distributed and precise representation enables the derivation of a local learning rule for the slow connections. The slow connections are responsible for the generation of the temporal dynamics of the output. The learning rule for these connections is simple. It is proportional to the firing

rate of the presynaptic neuron and to the error feedback received as a current by the postsynaptic neuron. In all these networks, the stochasticity in spiking is not a signature of noise but of the degeneracy of the representation. Since many neurons encode for the same feature, many spike patterns will result in the same output and precision. Therefore, even with a very small noise in the network, the spike trains will appear completely random and will be irreproducible form a trial to trial. Indeed, this work gives a radically different interpretation of the irregularity found in spike trains. Here, the randomness is not a signature of noise but on the contrary of precision and efficiency. Moreover, because of their efficient encoding, these networks do not need to be high dimensional to achieve a certain function as opposed to other approaches such as reservoir computing.

Contents

Chapter 1

Introduction

How does the brain learn to represent and process information? This question naturally unfolds into two fundamental problems: representation and learning. Investigating representations consists in understanding the neural code and how information is represented across neurons. The apparently noisy and unreliable nature of the neurons' output, in addition to the highly distributed nature of the neural code, makes it difficult to extract the exact nature of these representations. Neural representations are not, however, hard-wired but are learned from the stimuli we experience. Indeed, the brain undergoes constant learning to optimize our behavior and our representations of the world. Thus, understanding the ongoing learning that takes place in neural circuits is crucial to understanding the neural code. Synaptic plasticity is believed to be at the basis of these adaptation mechanisms. However, a large variety of plasticity rules is found in the brain, and understanding how these plasticity rules interplay to contribute to neural functions is very difficult. Here we present a novel theory that bridges together aspects of learning, synaptic plasticity and neural representations.

To introduce our problem we start by reviewing the common issues related to neural coding. We then present the balance between excitation, a phenomenon that is central to our theory and believed to play a major role in neural processing and coding. Next, we present recurrent neural networks that are used to model the computations in the brain. We finally review aspects of synaptic plasticity that are relevant to our subject.

1.1 Neural Coding

In many cortical areas, spike times are irregular and unreliable, with neurons responding to the same stimulus by spiking at different times on each trial (Tolhurst et al., 1983; Buracas et al., 1998). Following this observation, many assume that in these areas, information is conveyed through firing

rates, spikes being only noisy samples from the underlying rate. In this framework, spiking is usually modeled using a non-homogeneous Poisson process. Accordingly, the spikes of a single neuron are fired independently form one another. Experimental results show that taking into account the correlation between spikes does not increase the amount of information about the stimulus. However, in other areas, spike timing appears to be critical for coding (Theunissen and Miller, 1995; Gollisch and Meister, 2008; Bair and Koch, 1996; Buracas et al., 1998; Richmond et al., 1990, 1987; McClurkin et al., 1991). The term rate coding is commonly used when spike times are not critical to convey information about the stimulus. When the opposite occurs, the coding strategy is qualified as temporal.

Whether the coding strategy is temporal or rate-based, information is likely to be encoded by a large number of neurons (Deadwyler and Hampson, 1997; deCharms, 1998; Georgopoulos et al., 1986; Knudsen et al., 1987; Zohary et al., 1994). Thus, it is crucial to determine how neurons collaborate to represent relevant information. A common approach is to consider that neurons encode information independently from one another (Georgopoulos, 1990; Schwartz, 1994). This does not however exclude the pooling of activity of many independent neurons to obtain more accurate estimates of variables. It only means that neurons "ignore" each other when they represent information. For example, a population that combines independence across neurons, together with Poisson firing statistics, is said to employ an independent Poisson code. Since information is carried by the rate of the neurons, determining this rate is crucial to extract the encoded variables. Indeed, rate estimates determined from the activity of one neuron in a short time bin are noisy and unreliable. However, if all the neurons in the population fire at the same rate, averaging the activity of many neurons at once will result in precise estimates. Specifically, the error on the rate estimate in such a scenario scales $1/\sqrt{N}$ where N is the number of neurons (Shadlen and Newsome, 1998, 1994). Many decoding schemes were developed using the assumption of independent coding (Brunel and Nadal, 1998). For example, in the population vector approach, neurons have widely distributed tuning curves that depend on the stimulus value, with each peaking at a different location. One can infer a stimulus value by combining the activity of the different neurons in a Bayesian optimal way (Pouget et al., 1998; Zemel et al., 1998; Deneve et al., 1999).

Indeed, the independent coding hypothesis is suitable for theoretical analysis. However, in some areas neurons have been found to coordinate their activity, and information may be contained in the relative timing between spikes of different neurons (Gray and McCormick, 1996; Abeles et al., 1993). For example, hippocampal place cells signal the animal location by

firing at particular times relative to the ongoing oscillation phase in the population (O'Keefe and Recce, 1993). Other studies have found that in the auditory cortex, neurons synchronize their spiking following the onset of a stimulus while keeping their individual firing rates constant (deCharms and Merzenich, 1996). On a more speculative level, the lower bound to the error induced by discretizing a continuous signal using N spikes is $1/N$, which is considerably lower than the $1/\sqrt{N}$ of an independent Poisson code. Thus, to achieve a certain level of precision, a population employing an independent code needs a large number of spikes, which can be seen as a lack of efficiency (Deneve and Machens, 2016).

How can a code with apparent spike randomness be efficient and achieve high accuracy while using few spikes? What type of coordination between neurons does such a code require?

1.2 E-I Balance

1.2.1 Theoretical Foundation

What are the mechanisms that underlie the randomness in spike trains? (Mainen and Sejnowski, 1995) shows that when a constant current is injected into a neuron, it responds by spiking at regular intervals. This means that the stochasticity in spiking does not arise solely form internal sources of noise in the neuron. In an attempt to understand the origins of the irregularity in spiking, two ideas were proposed.

The first idea assumes that because spike trains are irregular, neurons must act as coincidence detectors that convey precise temporal patterns of spike arrivals (Softky and Koch, 1993). In this scheme, presynaptic spikes arriving from different neurons must occur simultaneously in order to bring the membrane potential of the post-synaptic neuron to the spiking threshold. Otherwise the membrane leak will inevitably bring the voltage to rest. However, because of the overwhelming number of synaptic inputs a neuron receives in the neocortex, detecting these events requires a very short membrane time constant. The required time constant is much shorter than the membrane time constant observed in biological neurons. However, detecting such coincidences with time constants comparable to the one found in the brain is possible if spiking events are rare, as observed in some subcortical areas.

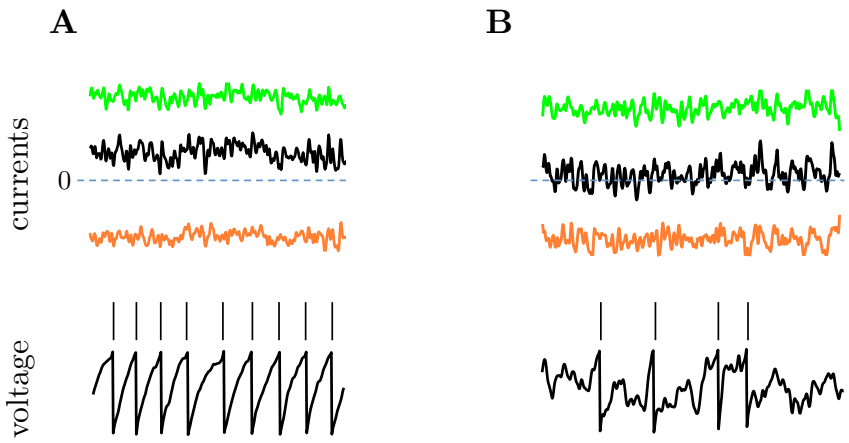

Figure 1.1 – Balance between excitation and inhibition produces irregular spike trains. **A**: upper panel: the neuron receives an excitatory input (green curve) stronger than the inhibitory input (orange curve). The total current is represented by the black line and has a positive mean. Bottom panel: the voltage of the neuron systematically drifts towards the threshold resulting in a regular spike train. **B**: When the excitatory and inhibitory currents cancel each other on average, the membrane potential undergoes a random walk towards the threshold resulting in irregular spikes.

As seen previously, many assume that the variable Interspike Interval (ISI) that is observed in the spike train of a neuron does not reflect the transmission of precisely timed coincidence events, but on the contrary, of a noisy rate code. How can such a noisy code arise from the integration of thousands of synaptic inputs that are mainly excitatory? Normally, such a high conductance regime would make the neuron fire very regularly, transmitting regular spike trains to next layers (fig. 1.1A). Inspired by a random walk model initially proposed by (Gerstein and Mandelbrot, 1964),(Shadlen and Newsome, 1994) explains this randomness by a balance between the excitation and the inhibition received by the neuron. The idea behind the model is simple: the membrane potential performs a random walk to the spiking threshold under the action of excitatory and inhibitory inputs. Excitatory inputs push the voltage towards the spiking threshold, and the inhibitory inputs drive it away from such threshold. If the neuron receives equal amounts of excitation and inhibition, the times at which the neuron reaches the spiking thresholds are random (fig. 1.1B). Using a model where an integrate-and-fire neuron receives a big number of excitatory and inhibitory irregular spike trains, (Shadlen and Newsome, 1998) showed that the output spikes of the neurons were highly irregular and conserved the same statistics as the input spike trains. Moreover, such a mechanism works with the realistic membrane time constants that are

found in the brain. However, the initial studies were conducted on a single neuron receiving excitatory and inhibitory input. Later studies show that this balance of excitation and inhibition emerges also in recurrent networks undergoing few constraints.

1.2.2 Balanced Recurrent Networks

What are the conditions to have a balance between excitation and inhibition in a network? Counterintuitively, such a balanced state does not necessarily require a fine-tuning of the connections. For example, it is an emergent property in sparsely and randomly connected recurrent networks(van Vreeswijk and Sompolinsky, 1996, 1998; Amit and Brunel, 1997; Brunel, 2000). In this context, sparse means that the number of connections a neuron receives $K << N$, the number of neurons in the network. In these networks, excitation and inhibition cancel each other dynamically: if excitation increases, it will lead to an increase in the inhibitory firing rate, thus balancing the excitatory input to the neurons. As described in the previous section, the membrane potential undergoes a random walk resulting in irregular firing. The fluctuations of the membrane potential are proportional to the mean input, resulting in firing rates that are proportional to the input. The neurons in the network thus follow a rate-coding strategy. Moreover, owing to the sparse connectivity, two neurons are unlikely to receive inputs from the same group of neurons, resulting in uncorrelated membranes potential and activity between them. Such a network typically uses an independent Poisson-like code.

1.2.3 Experimental Evidence of Balance

At this point, the idea of Balance between excitation and inhibition is very appealing from a theoretical point of view but lacked substantial experimental evidence. However, this was not in contradiction with some experimental observations (Shadlen and Newsome, 1994). (Borg-Graham et al., 1996; Anderson et al., 2000; Monier et al., 2008) developed a method to measure the excitatory and inhibitory conductance in vivo. This method consists of clamping the membrane potential at different values and injecting a current into the neuron. They then fit the following simple conductance-based equation to the values of V and I_{inj}

$$C\frac{dV}{dt} = -G_{leak}(V - V_{leak}) - G_{ext}(V - V_{ext}) - G_{inh}(V - V_{inh}) + I_{inj} \quad (1.1)$$

where C is the capacitance of the membrane, V is the membrane potential of the neuron, G_{leak}, G_{ext} and G_{in} are respectively the leak, the

excitatory and the inhibitory conductances. V_{leak} is the resting potential of the neurons and V_{ext} and V_{linh} are respectively the excitatory and inhibitory reversal potentials. I_{inj} is the injected current. Specifically, the global conductance is inferred at first and is then decomposed as excitatory and inhibitory depending on the value of the membrane potential and the reversal potentials (Monier et al., 2008). A balance exists if the relationship between both conductances is linear across conditions, that is, excitatory and inhibitory conductance are proportional. Indeed, such proportionality was found in slices of ferrets' prefrontal cortex during spontaneous activity (Shu et al., 2003). The ratio between the conductances is stable for many neurons and tends to cluster around 1. This relationship was also confirmed in vivo in anesthetized ferrets (Haider et al., 2006). Balance was also found during stimulus-evoked activity. For example, in auditory cortex of anesthetized rats, excitation and inhibition were co-tuned and peaked for the same value (Wehr and Zador, 2003). However, such a co-tuning is not universal. In other areas, inhibition was found to be more broadly tuned (Wu et al., 2008; Cardin et al., 2007; Kerlin et al., 2010), but the strongest response for excitation and inhibition to a neuron was always evoked by the same stimulus.

The method described previously only allows the measurement of either excitation or inhibition at one time in a single neuron. For example, in voltage clamp experiments, the type of conductance that is measured depends on the value of the clamped membrane potential. It is thus impossible to measure both conductances in a single trial. However, such a measurement is critical in order to determine the temporal structure of balance, e.g. whether excitation and inhibition are tightly correlated or simply compensate on average. (Okun and Lampl, 2008) bypasses this problem by measuring excitatory and inhibitory drives in nearby neurons. In fact, and as shown by the study, such neurons receive extremely correlated inputs. Measuring excitation received by a neuron is therefore similar to measuring it in a nearby neuron. It was found that excitation and inhibition are highly correlated during spontaneous activity and stimulus-evoked activity (whisker deflection). (Xue et al., 2014) also confirmed a tight balance on a 1ms level.

Balance between excitation and inhibition was first hypothesized to explain the irregular ISI in cortical spike trains (Shadlen and Newsome, 1994, 1998). Such an irregularity seems to only require a balance on average, the excitatory and inhibitory inputs being uncorrelated on a finer time-scale. The experimental findings presented previously indicate, however, that cortical balance is regulated on finer time scales. What could be the computational role of such a tight balance beyond irregular spiking?

1.3 Computations

The brain is able to perform very complex information processing. Even very simple tasks require a multitude of operations such as storage, recognition and extraction of relevant information, decision-making and generation of motor commands. Indeed, these computations are supported by the dynamics of neural circuits. To understand and model how the neurons collectively perform these computations, one usually first describes the task in terms of input/output transformation. For example, in a decision-making task, the input is the stimulus and the output is the choice performed by the subject. The relationship between the input (stimulus) and output (response) is commonly described using dynamical systems or differential equations. A network is then constructed to implement this relationship.

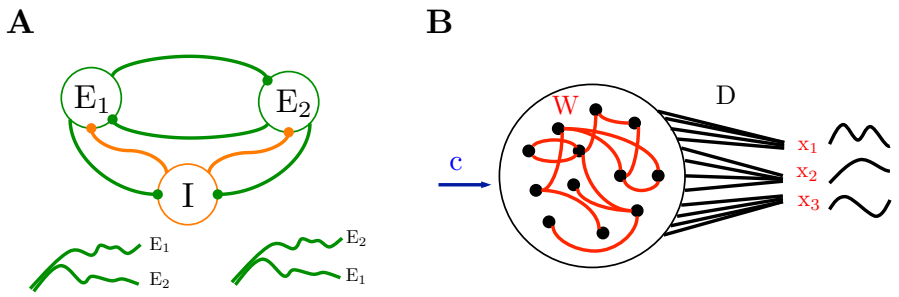

Figure 1.2 – specific v. generic networks. **A**: a decision making network has two excitatory populations competing through inhibitory interneurons. The population that has the highest rate indicates the choice of the monkey regarding the motion of the dots. The two bottom panels represent the only two attractor states in this network. **B**: A generic recurrent network. The network receives a command c and produces dynamic outputs x_1, x_2 and x_3. To optimize the output of the network, one should learn the recurrent and/or the readout weights **W** and **D** respectively.

1.3.1 Handcrafted v. Generic Networks

To implement a particular task, one can construct a specific network that qualitatively fits the behavior of neural circuits. In a motion discrimination task, for example, a monkey should decide in which direction a group of points are moving (Britten et al., 1992). The amount of noise added to the stimulus in the form of randomly moving points can make the task easier or harder. Single unit neural recordings in the prefrontal cortex show that the firing of certain neurons is highly correlated with the choice of the monkey. This task is modeled using two excitatory populations competing via an inhibitory population (Wang, 2002). The network exhibits only two stable

patterns of activity with every time only one population firing at high rate (fig. 1.2A). The choice made by the network is determined dependent on which population is more active. Such a network is called an attractor network because, under the influence of the stimulus, it exhibits particular patterns of activity where the dynamics of the network gets trapped. The attractor network framework is successful in modeling a variety of behaviors and reproduces aspects of the neural data. However, these networks are usually handcrafted and the connections have to be set by hand in order to reproduce the desired behavior.

The previous approach is mainly used to fit relatively simple neural behavior. It is, however, much more difficult to use it to construct networks that preform complex and dynamic computations like those involved in sensorimotor control. A solution to this problem is to use a generic recurrent neural network (RNN) that learns the relevant behavior, using some optimization or learning procedure. The information is extracted from the network with readout neurons that decode the activity of neurons (fig. 1.2B). One typically optimizes the readout and/or the recurrent weight in order to produce the desired output. Since we are interested in dynamic and highly precise computations we will only focus on this approach. Next we will present several models that use this approach.

1.3.2 Optimizing the Readout Weights

In the early 2000s, a new approach known as liquid or "reservoir" computing, which is radically different from the attractor approach was proposed (Maass et al., 2002; Jaeger, 2001). It is more suitable for real-time computing and dealing with rapidly varying and dynamic inputs.

Instead of using a specific architecture dedicated to a particular task, this method uses a generic randomly connected recurrent network commonly called the reservoir. The network acts as a non-linear projection of the input in the high dimensional space. At a certain point in time, the activities in this network are not only a momentary projection of the actual input, but also reflect the history of the past inputs due to the reverberation in the network mediated by the recurrent connectivity. Such memory facilitates any further temporal processing that can eventually be performed by the next layer. Indeed, a readout layer will then combine linearly the activity of the neurons in the reservoir and thus perform any wanted discrimination, categorization, transformation based on the input.

To learn a particular discrimination task or an input/output transformation, only the readout weights, i.e., the ones from the RNN to the read

out neurons, are modified. Because the learning only takes place at the output layer, a single reservoir can be used to learn different tasks simultaneously. In fact, it only serves as a general prepossessing tool projecting signals into a high dimensional non-orthogonal basis. To learn an additional task, one only needs to add a new set of readout neurons.

This approach is also fundamentally different from the attractor framework because it does not rely on any stable states to perform computations. On the contrary, it relies on the transient dynamics and on the near chaotic behavior of the RNN. In fact, the reservoir must be sensitive to changes in the input, so that small changes can induce different trajectories in the reservoir. This is called the separation property and it is a necessary condition for the functioning of the model.

Even though this method is efficient at learning complex and highly non-linear input/output relationship, it is mainly criticized for being a black-box approach. If it allows the reproduction or implementation of complex functions, it does not, however, permit an understanding of how a neuron precisely computes and the type of representation they bare. In addition, to obtain the separation and the approximation properties that are required, the reservoir must contain many neurons. This generates more activity than needed, which can be seen as inefficient. Learning the recurrent weights may be a solution to increase efficiency. Optimizing the recurrent weights may make a better use of the resources and thus require networks with lower dimensions.

1.3.3 Optimizing the Recurrent Weights

Learning recurrent weights is very challenging because of the long temporal dependencies that may be induced by recurrence. This is believed to be the main reason why first order methods such as back-propagation through time (BPTT) (Rumelhart et al., 1986) fail in many scenarios. Briefly, BBTT consists of unfolding the temporal dynamics of the network and translating it into a multi-layered perceptron where each layer is an instant in time. One then applies the regular backpropagation algorithm. However, in recurrent networks, the error signal may decay or grow exponentially. Thus the error signal can be overwhelmed by reverberation, which makes the dependence of the error on the weights hard to assess as soon as the target is separated from the input by several time steps (Bengio et al., 1994). A new method called Hessian-free optimization (Martens, 2010; Martens and Sutskever, 2011) brings a solution to this problem. However, the efficiency of this method highly depends on the initialization. Some initial configuration may lead overwhelmingly exponential growth in errors, thus

destabilizing the algorithm. This is particularly problematic for recurrent networks that produce cortex-like irregular spiking. These networks are chaotic and are highly sensitive to small changes in the inputs.

Force Learning

FORCE Learning (Sussillo and Abbott, 2009), inspired by reservoir computing bypasses these stability issues by using a feedback loop that suppresses chaos in the network. The suppression of chaos relies on a learning rule that performs strong and fast modification of the weights. The network succeeds in learning to generate complex output patterns in the presence or absence of an input to the network. This method does not only apply to readout weights, it also applies to recurrent weights. The scheme initially developed for rate network seems to be problematic when applied as is to spiking models. However, (Thalmeier et al., 2015) succeeded in translating force learning to spiking recurrent networks using a predictive coding approach. The success of this method resides in the inherent rate description that underlies the spiking dynamics. This approach is particularly relevant to our work and will be explained in more detail at the end of chapter 4.

Learning by Copying an Auto-Encoder

(DePasquale et al., 2016) propose an alternative approach to train the recurrent weights in a spiking network. A notable difficulty in training recurrent weights is the absence of a specific target to guide the optimization. Optimizing the readout weight is straightforward because there is a direct relation between the weights and the desired output. For example, if the decoding is linear, the output of the network is $\hat{x}(t) = \sum_n D_n a_n(t)$ where $a_n(t)$ is the activity of neuron n. Since the decoded output should be equal to the desired output, one can directly optimize those weights using a least squares procedure. Such an optimization is however not straightforward for the recurrent weights because there no direct link between the weights and the desired output.

Indeed the recurrent weights have a direct influence on the activity of the neurons. What pattern of activity in the network can produce the target output? Once this target activity is determined the recurrent weights are optimized to produce it. To determine the desired target activity, the approach uses an auxiliary auto-encoder of rate units. In fact, the desired output is fed as an input to this randomly connected auxiliary network. The activity generated by this auto-encoder can be combined by the decoder to approximate the target output. It is this activity that serves as a target for the spiking units in the original network. In these networks the precision of the code is still proportional to $1/\sqrt{N}$ as opposed to the

approach used by Thalmeier et al. (2015) which employs a $1/N$ precise code.

The previous methods are efficient in constructing networks that perform useful computations. However the learning techniques used to optimize them are non-local. For example, force learning requires inverting correlation matrices of the rate in the network in order to assign a learning rate to each neuron. In the case of spiking rate-based networks, the optimization requires the use of an external teaching signal (the activities of the rate network) in an unnatural way. Indeed, the brain only relies on local plasticity rules to learn from and adapt to its environment. How can functional recurrent networks learn using only local plasticity rules such as those exhibited by the neural circuits?

1.4 Plasticity

The brain is capable of adapting to a world that evolves continuously. This ability is ensured by the plasticity of cortical circuits that change to optimize their output based on external inputs, their own activity or other modulation signals. For example, on a sensory processing level, the sharpening of neural responses during development or the emergence of receptive fields is a learning process influenced by the stimuli the brain receives (Freedman et al., 2006). Beyond sensory representation, learning is also involved in more dynamic computations. For example, sensorimotor control uses internal models to predict the temporal trajectory of effectors. The features of the effectors may change over time. Internal models representing the dynamics of these effectors become erroneous and should therefore be updated (Kording et al., 2007). A neural network that implements such internal models should be able to update its implementation in order to perform accurate predictions.

It is widely believed that learning in the brain is supported by the modification of synaptic efficacy between neurons. Indeed, from a modeling point of view, changing the connection weights in a network highly influences the neurons' dynamics and responses. More or less direct experimental evidence have corroborated this point of view (Mehta et al., 1997; Zhang et al., 1998; Engert et al., 2002; Feldman and Brecht, 2005). In addition, on a cellular level, experimentalists can directly observe and induce synapse weakening and strengthening known as long term depression (LTD) and potentiation (LTP), respectively (Bliss and Lomo, 1973). These forms of plasticity can be induced by jointly monitoring the activity of the pre- and postsynaptic cells. The resulting change in synaptic strength can last from minutes to hours. For example, presynaptic stim-

ulation in conjunction with strong or weak depolarization induces LTP or LTD respectively (Artola et al., 1990). Here we are mostly interested in functional implications of plasticity. In other words, we want to know how the rules that govern LTD and LTP relate to coding and computations.

In this section we will first consider Hebbian learning, a hypothesis that had great conceptual influence on the study of plasticity both from a theoretical and experimental point of view. We will then present the factors that influence LTD and LTP in biological neurons. Finally, we will review theoretical work that investigates the functional role and the computation principles that underlie synaptic plasticity.

1.4.1 Hebbian Learning

Hebb's Hypothesis

(Hebb, 1949) proposed the following hypothesis:

"When an axon of cell A is near enough to excite B and takes repeatedly hand in firing it, some growth process or metabolic changes take in one or both cells that such A's efficiency, as one of cells firing B, is increased"

This idea was confirmed later by several experimental findings where LTP and LTD were found to depend on the joint activities of the pre- and post-synaptic neuron. According to Hebb's idea, changing the efficiency of the connection between the neurons depends on the correlation of the activities of both neurons. It is obvious how this idea relates to stimulus conditioning experiments. If a stimulus A is Followed by stimulus B repeatedly, the respectively selective neurons, say N_A and N_B, will also fire in this order, resulting in strengthening of connections from N_A to N_B. Thus, if A is presenting alone, the firing of N_A will result in the firing N_B even in the absence of stimulus B.

Most importantly, the interest of this rule relies on its locality. It only relies on local information to modify the synapse, which is a crucial property in modeling synaptic plasticity. In fact, a synapse has *a priori* only access to local information.

Rate-Based Hebbian Learning

Mathematically the previous idea was translated into:

$$\Delta w_{ij} \propto r_i r_j \tag{1.2}$$

Where w_{ij} is the weight of the connection between i and j and r_i and r_j are the activities/output of neuron i and j respectively. This model usually assumes that activities are akin to firing rates and are thus continuous.

However, pure Hebbian learning has two main problems: instability and absence of competition. First, if the activities between neurons are correlated it will induce a strengthening of their connection, which will in turn increase their correlations. This causes unbounded growth of their synapses. Second, since the Hebbian learning rule updates a weight independently from all the others, this results in a lack of competition between synapses. Competition between synapse is a desirable feature in plasticity because it can lead to the emergence of selectivity in neurons.

Synaptic normalization is a possible alternative to limit the uncontrolled growth of the weights. It forces the weight vector of a neuron to have a constant norm. We illustrate this idea using a single postsynaptic neuron that receives J inputs x_j with weights w_j and produces an output y. For each synapse the pre-synaptic activity is x_j and the post-synaptic activity is y. Thus the normalized weight after update is:

$$w_i(t+1) \propto \frac{w_i(t) + x_i y}{(\sum_k^N w_j(t) + \epsilon x_j y)^2} \tag{1.3}$$

The regular Hebbian update is devised by the norm of the weight vector after the update. However, such a rule requires for each synapse the knowledge of all the other synaptic weights of this neuron, which violates the locality constraint. Using a Taylor expansion and some approximations, the previous rule can be turned to a local learning rule known as Oja's rule (Oja, 1982) :

$$\frac{dw_i(t)}{dt} \propto \epsilon y(x_i - w_i(t)y) \tag{1.4}$$

One can show that the final weight vector $\mathbf{w}$ is aligned with the first principle component of $\mathbf{x}$. This is an example of a representational learning based on the statistical properties of the input. Because the final weight vector is in the direction for $\mathbf{x}$ that is the most variable, it thus extracts the most informative component of the signal. This constitutes an efficient representation and compression of the signal. Other types of normalization exist such as subtractive normalization, which fixes the total sum of the weights to a certain value. The exact nature of these constraints influences considerably the behavior of the system.

The BCM rule (Bienenstock et al., 1982) provides a different solution to control weight growth and induces competition between synapses. It

uses a variable threshold, Θ, to induce potentiation or depression. If the postsynaptic activity y is above the threshold the weight is potentiated and when the opposite occurs it is depressed. Thus, the weight dynamics are:

$$\frac{dw_i(t)}{dt} \propto \epsilon x_i(y - \Theta(t)) \tag{1.5}$$

The threshold dynamics should be super linear in the postsynaptic activity y to stabilize weight growth. For example a quadratic dependence on the average postsynaptic activity translates $\frac{d\Theta(t)}{dt} = \langle y^2 \rangle$. The super linear dependence is necessary to ensure that any high postsynaptic activity is quickly taken into account by increasing the threshold and thus depressing the synapses preventing uncontrolled growth loop. This also induces competition because the synapses compete to control the postsynaptic activity. This rule was successful in modeling the emergence of visual receptive field (Blais et al., 1998).

The learning rules presented here are mainly expressed in terms of firing rates. But as we will see next, synaptic plasticity has been also found to highly depend on spike times and many other factors.

1.4.2 Timing and the Variety of Factors that Influence Plasticity

Indeed, the first studies on LTP and LTD have primarily proved their dependence on the firing rates (Bliss and Lomo, 1973). However, a series of influential experiments conducted in the last two decades have consistently found a high dependence of plasticity on spike timing (Bi and Poo, 1998; Markram et al., 1997; Bell et al., 1997). Such mode is called Spike Timing Dependent Plasticity (STDP). An extensively studied form of STDP is when synapses are potentiated if a presynaptic spike precedes a postsynaptic spike immediately and is depressed in the opposite case (fig. 1.3A) (Bi and Poo, 1998; Celikel et al., 2004; Markram et al., 1997). This form is consistent with the Hebbian hypothesis. In fact, a pre-post order can be seen as a causality link, implying that the synapse should be strengthened, while the post-pre order translates as an absence of such a link and thus the synapse should be weakened. For this reason this form of STDP is called Hebbian. Hebbian-STDP has been originally predicted by (Gerstner et al., 1996) in an attempt to model the emergence of selectivity to sound location in barn owl.

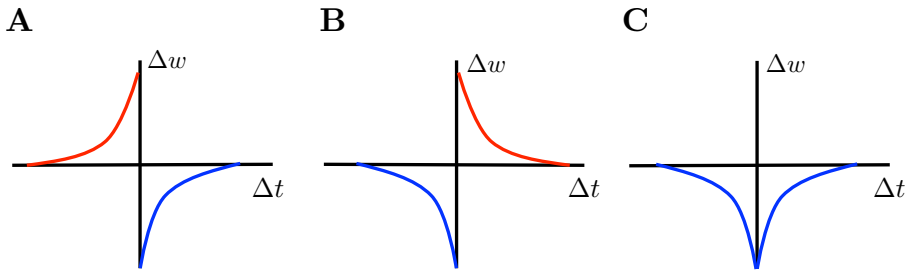

Figure 1.3 – STDP shapes. This figure shows three different STDP shapes found in slice experiments. It shows the weight change Δw as a function of the difference between the times of the presynaptic and postsynaptics spikes, $\Delta t = t_{pre} - t_{post}$. **A**: classical STDP. if the pre spike precedes the post spike there is potentiation and if the opposite occurs there is depression. **B**: Anti-hebbian STDP has an inverse profile of the classical STDP. **C**: Anti-Hebbian LTD where both orders result in depression.

Other shapes of STDP are also exhibited at other synapses. for example, some rules exhibit an inversed profile of the Hebbian type, where the pre-post order results in weakening and post-pre order leads to the opposite (fig. 1.3B). Consequently, it is called anti-Hebbian STDP (Fino et al., 2005, 2008). In most cases the LTP branch is completely absent and both spiking orders result in weakening (fig. 1.3C) (Han et al., 2000; Requarth and Sawtell, 2011; Lu et al., 2007). In all cases the amplitude of change is the highest when the two spikes are contiguous (10 ms) (Sjostrom et al., 2001; Markram et al., 1997). The change decreases with the time distance and becomes nonexistent above a certain lag. Most of these experiments are realized on slices where the background activity can be tightly controlled.

If first thought to be a universal and fundamental kernel for plasticity, basic STDP seem to depend on many factors. For example, *in vivo*, the presynaptic and postsynaptic activities do not consist of regularly repeated pairs of pre-post spikes, but rather of irregular spike trains. In this case the total change in the synaptic weight cannot be inferred by linearly summing the weight changes induced by each pre-post pair (Wang et al., 2005; Wittenberg and Wang, 2006). In fact, experiments show non-linear interactions between triplets and quadruplets of spikes (Froemke and Dan, 2002). STDP is also influenced by rate. For example, to induce LTP, the pairing should be repeated at a certain frequency. In addition, above a certain frequency LTP does not depend anymore on the temporal order of pre/post spikes (Sjostrom et al., 2001). Moreover STDP is sensitive to the subthreshold depolarization just after pre/postsynaptic spike(Sjostrom et al., 2001). Furthermore, STDP seems to be subject to modulation. For ex-

ample, in some synapses, Anti-Hebbian LTD can be modified into Hebbian STDP by manipulating depolarization on dendrites (Sjöström and Häusser, 2006).

Moreover, spike timing seems to be a factor among others for plasticity. In some experiments LTP/LTD is induced without a postsynaptic action potential. A strong postsynaptic sub-threshold depolarization potentiates the synapse, and a weak depolarization depresses it (Artola et al., 1990; Ngezahayo et al., 2000). This led to a debate whether STDP is less fundamental than voltage-based plasticity (Lisman and Spruston, 2005). In fact, since spikes are particular events where the voltage peaks briefly, some argue that STDP can be inferred form voltage based plasticity(Clopath et al., 2010). In any case, depolarization, with or without a spike, seems to be a key factor for LTP/LTD. Depolarization at dendrites controls the opening of NMDA receptors and thus the influx of calcium into the neuron. High calcium concentration is thought to trigger processes that lead to LTP and low calcium concentration leads to LTD (Yang et al., 1999).

All these studies imply that LTD and LTP are highly malleable and are influenced by a multitude of factors such as spike timing, firing rate, and depolarization. How does these dependencies relate to function? More precisely what are the computational principles that shape the dependencies of long term plasticity.

1.4.3 Plasticity and Functions

Bottom up Approaches

To understand the functional underpinnings of plasticity, modelers usually apply rules that are abstracted from experience and study their effects in neural networks. (Song et al., 2000) simulated a Hebbian-STDP in a neuron that receives 1000 synapses (Inhibitory and Excitatory). The kernel used to model LTP and LTP is constituted of two exponentials with amplitude A_+ , A_- and time constants τ_+ and τ_- for LTP and LTD respectively. Thus, the plasticity kernel is:

$$\Delta w = \begin{cases} A_+ e^{\frac{\Delta t}{\tau_+}} & \text{if } \Delta t < 0 \\ A_- e^{\frac{-\Delta t}{\tau_-}} & \text{if } \Delta t > 0 \end{cases} \tag{1.6}$$

The weight change is driven by $\Delta t = t_{pre} - t_{post}$, the time difference between the presynaptic and the postsynaptic spikes. They found that this rule leads to a stable configuration of weights and rates without the addition of any ad-hoc stabilization term like in rate-based Hebbian learning. This effect was also studied in (Van Rossum et al., 2000; Kempter et al.,

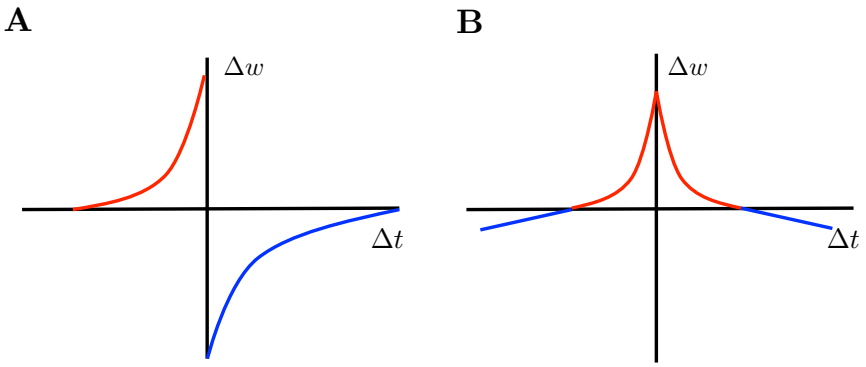

Figure 1.4 – STDP in theory. **A**: Classical STDP where LTD dominates. The total area under the curves is negative. **B**: speculated STDP that leads to detailed excitatory and inhibitory balance. The weight update does not depend on spike order but on the absolute value of the difference between the spike times. If the spikes are contiguous the weight is increased and when they are distant the weight is decreased.

2001). Nevertheless, a condition for this stability is that the integral of the kernel should be negative, meaning that there is a partial bias towards LTD (fig. 1.4A). Some experimentally found STDP's exhibited this asymmetry (Celikel et al., 2004; Froemke et al., 2005). Moreover, they found that STDP also enforces a balance between excitation and inhibition rendering the postsynaptic spike train irregular. The ability of inhibitory STDP to regulate E/I balance has been particularly studied in (Vogels et al., 2011). Instead of Hebbian-STDP, they speculated a kernel that does not depend on spiking order but on the time difference between pre and post spikes (fig. 1.4B). For short lags it induces LTP and for longer lags it induces LTD. It acts on inhibitory synapses only, excitatory synapses being fixed and random. The rule is first studied in the case of a single neuron receiving excitatory and inhibitory inputs and is then applied in a recurrent network. They show how by enforcing balance, the plasticity rule improves the coding properties in the single neuron model and drives the recurrent network into the asynchronous irregular regime.

As discussed previously, plasticity does not solely depend on spike times, but is the result of complex interactions between rates, spike times, and depolarization. Models that only depend on spike pairs and their timing fail to capture many experimental findings. A number of models tried to capture and integrate this multitude of factors into a single model. While (Shouval et al., 2002) uses a mechanistic model, (Clopath et al., 2010) proposes a phenomenological approach that uses a simpler model more suitable for analysis in networks. In the latter, weight changes depend on

the filtered presynaptic spikes and the filtered postsynaptic membrane potential, with two different thresholds for LTD and LTP. LTP or LTD is induced depending on the value of the filtered postsynaptic depolarization relatively to these thresholds. The model succeeds in reproducing a number of phenomena such as the spike timing dependence of Hebbian STDP or the non-linear interaction between triplets. When tested in small recurrent networks, it allowed the formation of bidirectional connectivity in some experimental setups as opposed to classical STDP, which only enforces asymmetric motifs. However, when neurons in the network are stimulated in fixed temporal order, the rule captures the temporal structure of the input like in common STDP (Rao and Sejnowski, 2001; Gerstner et al., 1996). Finally, It also allows the emergence of realistic receptive fields when the network is exposed to patches from natural images.

The majority of these studies focus on the effect of plasticity on the weight distribution and network dynamics. It is, however, difficult to relate these rules to computationally complex functions. For this reason, some use another approach where they start from functions and infer biologically plausible plasticity rules.

Normative Approaches

Normative approaches to plasticity use objective functions as a starting point rather than empirical plasticity rules. After defining an appropriate objective for the task, the learning rules are derived by optimizing this objective. Indeed, these methods are canonical in artificial intelligence. For example, the perceptron (Minsky and Papert, 1969), the multi-layered perceptron (Rumelhart et al., 1986) and support vector machines (Cortes and Vapnik, 1995) all employ rules that are derived using a gradient descent on some cleverly chosen objective function.

If these learning rules straightforwardly relate to functions, they may be however poorly grounded experimentally (Rumelhart et al., 1986). For example, they could suffer from biological implausibility, requiring processes that do not exist or are yet unknown in neurons. A way to remedy this is to approximate the derived rules with rules that are closer to biological reality but still satisfy the objectives. (Pfister et al., 2006) derives an optimal STDP kernel that teaches a neuron to generate a desired spike train in response to presynaptic spike patterns. (Gutig and Sompolinsky, 2006) derives a voltage dependent learning rule that enables a neuron to classify incoming spike train patterns from several neurons. More precisely, the neuron learns to decide whether to spike or not in response to incoming spike trains. (Toyoizumi et al., 2004) derive STDP window by maximizing

mutual information between presynaptic and postsynaptic neuron. Using the same technique, the authors also infer a spiking equivalent of the BCM rule (Toyoizumi et al., 2005). (Habenschuss et al., 2013; Kappel et al., 2014), derived optimal STDP that yields optimal decoding weights in a Bayesian inference framework (Jazayeri and Movshon, 2006). They then test a classical STDP and show that the learned weights had a similar structure to the one learned using the optimal rules.

Most of these studies consider mainly feedforward networks or a single neuron receiving multiple presynaptic inputs. The brain is, however, recurrently connected. Indeed, recurrence changes highly the dynamical properties of learning rules because it induces very long temporal correlations. Thus, what is guaranteed to work in a feedforward architecture may fail in recurrent networks.

1.4.4 Local learning in Recurrent Networks

One of the main obstacles for local learning in recurrent networks is the difficulty of inferring local learning rules from global objectives. Here we briefly present two studies where recurrent networks succeed in learning complex tasks using only local learning rules.

Using a bottom-up approach, (Lazar et al., 2009) is one of the few studies that successfully uses STDP to perform complex tasks. In a recurrent network of excitatory and inhibitory threshold units, the E to E connections undergoes STDP in addition to synaptic scaling and threshold adaptation. The network with plasticity is shown to be significantly more efficient than randomly connected reservoirs in performing complex counting tasks. However, such a bottom-up approach does not allow a deep understanding of why STDP enables such learning.

(Zylberberg et al., 2011) uses a top-down approach to learn a generative model of images with a recurrent network. The learning rules are derived from a single objective that contains an error measure, an L^o constraint on average firing rates (sparsity constraint), and a term that enforces low pairwise correlation of spike count between neurons (decorrelation constraint). The network undergoing these learning rules develops receptive fields like the ones in (Olshausen and Field, 1996).

1.5 Objectives and Organization

Three challenging neuroscience questions arise from this introduction:

1. Why do single neurons in large networks seem to fire at random?

2. What is the computational role of the tight balance between excitation and inhibition observed in the cortex?

3. How can a recurrent neural network learn to carry out a given function using local plasticity rules?

The first challenge concerns the random firing of individual neurons. This randomness is usually assumed to reflect noise, accumulated in the nervous system over many processing stages. Here we show that as a network learns to efficiently represent information, its neurons will eventually generate spike trains that appear to be completely random, even though the representation is essentially noise-free at the population level. Moreover, based on the highly redundant representations found in cortex, and the seemingly random firing of neurons, function is generally only assigned at the population level, e.g. (Pouget et al., 2000), single neurons being considered noisy and unreliable. Here we bring a radically different answer and show that if a network learns to represent information as precisely and parsimoniously as possible, specific functional meaning (in the form of coding errors) can be assigned at all levels, from the currents and voltages of single neurons to the whole population.

The second challenge concerns the balance between excitation and inhibition. This principle has been hypothesized to explain the variable spiking in the brain. The balance that is required to reproduce such variability is much looser than the balance found in some cortical areas. What could be the role of a tight balance? We show that tight balance is a signature of a highly coordinated spiking in the network which results in an efficient spike-based population code.

The third challenge is theoretical and concerns learning in recurrent neural networks. Most approaches to learning in neural networks assume so-called feedforward architectures. Indeed, all deep-learning approaches and their recent successes are built on such architectures (Hinton and Salakhutdinov, 2006; Silver et al., 2016). The main reason is that it has been extremely difficult to learn anything in recurrent neural architectures, and much less so if they consist of spiking neurons. To date, most approaches rely on specifying learning rules for single neurons, and then studying their effect on recurrent networks post-hoc, e.g. (Vogels et al., 2011; Gutig, 2016) or optimize the networks using non-local procedures (Sussillo and Abbott, 2009). Here we derive biologically realistic learning rules for a spiking recurrent neural network from first principles - efficient representation of

information - and provide an exact solution to the learning problem. In other words, learning in our network can be completely understood across all levels, from the single synapse and neuron to the emerging network function.

Indeed, it was previously shown that one can design spiking neural networks that generate efficient spiking codes (Boerlin et al., 2013). However this work has failed to answer the question of how biological networks could ever achieve the required, highly specific architecture. Here we propose a solution to this problem through learning and plasticity.

This thesis is based on articles. In the first two articles (Chapters two and three) we mainly study neural coding. More precisely, we study how a recurrent network of spiking neurons learns to efficiently represent its input using local and biologically plausible plasticity rules (Bourdoukan et al., 2012; Brendel et al., 2016). In chapter four we present a third article that studies more complex computations. We show how a network learns to implement a specific linear dynamical system (Bourdoukan and Denève, 2015). All the studies are based on the predictive coding framework developed in Sophie Deneve's Lab in collaboration with Christian Machens ' Lab.

Chapter 2

Voltage-Rate Based Plasticity

Learning optimal spike-based representations

Ralph Bourdoukan*
Group for Neural Theory
École Normale Supérieure
Paris, France
ralph.bourdoukan@ens.fr

David G.T. Barrett*
Group for Neural Theory
École Normale Supérieure
Paris, France
david.barrett@ens.fr

Christian K. Machens
Champalimaud Neuroscience Programme
Champalimaud Centre for the Unknown
Lisbon, Portugal
christian.machens@neuro.fchampalimaud.org

Sophie Denève
Group for Neural Theory
École Normale Supérieure
Paris, France
sophie.deneve@ens.fr

Abstract

How can neural networks learn to represent information optimally? We answer this question by deriving spiking dynamics and learning dynamics directly from a measure of network performance. We find that a network of integrate-and-fire neurons undergoing Hebbian plasticity can learn an optimal spike-based representation for a linear decoder. The learning rule acts to minimise the membrane potential magnitude, which can be interpreted as a representation error after learning. In this way, learning reduces the representation error and drives the network into a robust, balanced regime. The network becomes balanced because small representation errors correspond to small membrane potentials, which in turn results from a balance of excitation and inhibition. The representation is robust because neurons become self-correcting, only spiking if the representation error exceeds a threshold. Altogether, these results suggest that several observed features of cortical dynamics, such as excitatory-inhibitory balance, integrate-and-fire dynamics and Hebbian plasticity, are signatures of a robust, optimal spike-based code.

A central question in neuroscience is to understand how populations of neurons represent information and how they learn to do so. Usually, learning and information representation are treated as two different functions. From the outset, this separation seems like a good idea, as it reduces the problem into two smaller, more manageable chunks. Our approach, however, is to study these together. This allows us to treat learning and information representation as two sides of a single mechanism, operating at two different timescales.

Experimental work has given us several clues about the regime in which real networks operate in the brain. Some of the most prominent observations are: (a) high trial-to-trial variability—a neuron responds differently to repeated, identical inputs [1, 2]; (b) asynchronous firing at the network level—spike trains of different neurons are at most very weakly correlated [3, 4, 5]; (c) tight balance of excitation and inhibition—every excitatory input is met by an inhibitory input of equal or greater size [6, 7, 8] and (4) spike-timing-dependent plasticity (STDP)—the strength of synapses change as a function of presynaptic and postsynaptic spike times [9].

Previously, it has been shown that observations (a)–(c) can be understood as signatures of an optimal, spike-based code [10, 11]. The essential idea is to derive spiking dynamics from the assumption that neurons only fire if their spike improves information representation. Information in a network may

*Authors contributed equally

originate from several possible sources: external sensory input, external neural network input, or alternatively, it may originate within the network itself as a memory, or as a computation. Whatever the source, this initial assumption leads directly to the conclusion that a network of integrate-and-fire neurons can optimally represent a signal while exhibiting properties (a)–(c).

A major problem with this framework is that network connectivity must be completely specified *a priori*, and requires the tuning of N^2 parameters, where N is the number of neurons in the network. Although this is feasible mathematically, it is unclear how a real network could tune itself into this optimal regime. In this work, we solve this problem using a simple synaptic learning rule. The key insight is that the plasticity rule can be derived from the same basic principle as the spiking rule in the earlier work—namely, that any change should improve information representation.

Surprisingly, this can be achieved with a local, Hebbian learning rule, where synaptic plasticity is proportional to the product of presynaptic firing rates with post-synaptic membrane potentials. Spiking and synaptic plasticity then work hand in hand towards the same goal: the spiking of a neuron decreases the representation error on a fast time scale, thereby giving rise to the actual population representation; synaptic plasticity decreases the representation error on a slower time scale, thereby improving or maintaining the population representation. For a large set of initial connectivities and spiking dynamics, neural networks are driven into a balanced regime, where excitation and inhibition cancel each other and where spike trains are asynchronous and irregular. Furthermore, the learning rule that we derive reproduces the main features of STDP (property (d) above). In this way, a network can learn to represent information optimally, with synaptic, neural and network dynamics consistent with those observed experimentally.

1 Derivation of the learning rule for a single neuron

We begin by deriving a learning rule for a single neuron with an autapse (a self-connection) (Fig. 1A). Our approach is to derive synaptic dynamics for the autapse and spiking dynamics for the neuron such that the neuron learns to optimally represent a time-varying input signal. We will derive a learning rule for networks of neurons later, after we have developed the fundamental concepts for the single neuron case.

Our first step is to derive optimal spiking dynamics for the neuron, so that we have a target for our learning rule. We do this by making two simple assumptions [11]. First, we assume that the neuron can provide an estimate or read-out $\hat{x}(t)$ of a time-dependent signal $x(t)$ by filtering its spike train $o(t)$ as follows:

$$\dot{\hat{x}}(t) = -\hat{x}(t) + \Gamma o(t), \tag{1}$$

where Γ is a fixed read-out weight, which we will refer to as the neuron's "output kernel" and the spike train can be written as $o(t) = \sum_i \delta(t - t_i)$, where $\{t_i\}$ are the spike times. Next, we assume that the neuron only produces a spike if that spike improves the read-out, where we measure the read-out performance through a simple squared-error loss function:

$$L(t) = \big(x(t) - \hat{x}(t)\big)^2. \tag{2}$$

With these two assumptions, we can now derive optimal spiking dynamics. First, we observe that if the neuron produces an additional spike at time t, the read-out increases by Γ, and the loss function becomes $L(t|\text{spike}) = (x(t) - (\hat{x}(t) + \Gamma))^2$. This allows us to restate our spiking rule as follows: the neuron should only produce a spike if $L(t|\text{no spike}) > L(t|\text{spike})$, or $(x(t) - \hat{x}(t))^2 > (x(t) - (\hat{x}(t) + \Gamma))^2$. Now, squaring both sides of this inequality, defining $V(t) \equiv \Gamma(x(t) - \hat{x}(t))$ and defining $T \equiv \Gamma^2/2$ we find that the neuron should only spike if:

$$V(t) > T. \tag{3}$$

We interpret $V(t)$ to be the membrane potential of the neuron, and we interpret T as the spike threshold. This interpretation allows us to understand the membrane potential functionally: the voltage is proportional to a prediction error—the difference between the read-out $\hat{x}(t)$ and the actual signal $x(t)$. A spike is an error reduction mechanism—the neuron only spikes if the error exceeds the spike threshold. This is a greedy minimisation, in that the neuron fires a spike whenever that action decreases $L(t)$ without considering the future impact of that spike. Importantly, the neuron does not require direct access to the loss function $L(t)$.

To determine the membrane potential dynamics, we take the derivative of the voltage, which gives us $\dot{V} = \Gamma(\dot{x} - \dot{\hat{x}})$. (Here, and in the following, we will drop the time index for notational brevity.) Now, using Eqn. (1) we obtain $\dot{V} = \Gamma\dot{x} - \Gamma(-\hat{x} + \Gamma o) = -\Gamma(x - \hat{x}) + \Gamma(\dot{x} + x) - \Gamma^2 o$, so that:

$$\dot{V} = -V + \Gamma c - \Gamma^2 o, \tag{4}$$

where $c = \dot{x} + x$ is the neural input. This corresponds exactly to the dynamics of a leaky integrate-and-fire neuron with an inhibitory autapse[1] of strength Γ^2, and a feedforward connection strength Γ.

The dynamics and connectivity guarantee that a neuron spikes at just the right times to optimise the loss function (Fig. 1B). In addition, it is especially robust to noise of different forms, because of its error-correcting nature. If x is constant in time, the voltage will rise up to the threshold T at which point a spike is fired, adding a delta function to the spike train o at time t, thereby producing a read-out $\hat{x}$ that is closer to x and causing an instantaneous drop in the voltage through the autapse, by an amount $\Gamma^2 = 2T$, effectively resetting the voltage to $V = -T$.

We now have a target for learning—we know the connection strength that a neuron must have at the end of learning if it is to represent information optimally, for a linear read-out. We can use this target to derive synaptic dynamics that can learn an optimal representation from experience. Specifically, we consider an integrate-and-fire neuron with some arbitrary autapse strength ω. The dynamics of this neuron are given by

$$\dot{V} = -V + \Gamma c - \omega o. \tag{5}$$

This neuron will not produce the correct spike train for representing x through a linear read-out (Eqn. (1)) unless $\omega = \Gamma^2$.

Our goal is to derive a dynamical equation for the synapse ω so that the spike train becomes optimal. We do this by quantifying the loss that we are incurring by using the suboptimal strength, and then deriving a learning rule that minimises this loss with respect to ω. The loss function underlying the spiking dynamics determined by Eqn. (5) can be found by reversing the previous membrane potential analysis. First, we integrate the differential equation for V, assuming that ω changes on time scales much slower than the membrane potential. We obtain the following (formal) solution:

$$V = \Gamma x - \omega\bar{o}, \tag{6}$$

where $\bar{o}$ is determined by $\dot{\bar{o}} = -\bar{o} + o$. The solution to this latter equation is $\bar{o} = h * o$, a convolution of the spike train with the exponential kernel $h(\tau) = \theta(\tau)\exp(-\tau)$. As such, it is analogous to the instantaneous firing rate of the neuron.

Now, using Eqn. (6), and rewriting the read-out as $\hat{x} = \Gamma\bar{o}$, we obtain the loss incurred by the sub-optimal neuron,

$$L = (x - \hat{x})^2 = \frac{1}{\Gamma^2}\big(V^2 + 2(\omega - \Gamma^2)\bar{o} + (\omega - \Gamma^2)^2\bar{o}^2\big). \tag{7}$$

We observe that the last two terms of Eqn. (7) will vanish whenever $\omega = \Gamma^2$, i.e., when the optimal reset has been found. We can therefore simplify the problem by defining an alternative loss function,

$$L_V = \frac{1}{2}V^2, \tag{8}$$

which has the same minimum as the original loss ($V = 0$ or $x = \hat{x}$, compare Eqn. (2)), but yields a simpler learning algorithm. We can now calculate how changes to ω affect L_V:

$$\frac{\partial L_V}{\partial \omega} = V\frac{\partial V}{\partial \omega} = -V\bar{o} - V\omega\frac{\partial \bar{o}}{\partial \omega}. \tag{9}$$

We can ignore the last term in this equation (as we will show below). Finally, using simple gradient descent, we obtain a simple Hebbian-like synaptic plasticity rule:

$$\tau\dot{\omega} = -\frac{\partial L_V}{\partial \omega} = V\bar{o}, \tag{10}$$

where τ is the learning time constant.

[1]This contribution of the autapse can also be interpreted as the reset of an integrate-and-fire neuron. Later, when we generalise to networks of neurons, we shall employ this interpretation.

This synaptic learning rule is capable of learning the synaptic weight ω that minimises the difference between x and $\hat{x}$ (Fig. 1B). During learning, the synaptic weight changes in proportion to the post-synaptic voltage V and the pre-synaptic firing rate $\bar{o}$ (Fig. 1C). As such, this is a Hebbian learning rule. Of course, in this single neuron case, the pre-synaptic neuron and post-synaptic neuron are the same neuron. The synaptic weight gradually approaches its optimal value Γ^2. However, it never completely stabilises, because learning never stops as long as neurons are spiking. Instead, the synapse oscillates closely about the optimal value (Fig. 1D).

This is also a "greedy" learning rule, similar to the spiking rule, in that it seeks to minimise the error at each instant in time, without regard for the future impact of those changes. To demonstrate that the second term in Eqn. (5) can be neglected we note that the equations for V, $\bar{o}$, and ω define a system of coupled differential equations that can be solved analytically by integrating between spikes. This results in a simple recurrence relation for changes in ω from the i^{th} to the $(i+1)^{th}$ spike,

$$\omega_{i+1} = \omega_i + \frac{\omega_i(\omega_i - 2T)}{\tau(T - \Gamma c - \omega_i)}. \tag{11}$$

This iterative equation has a single stable fixed point at $\omega = 2T = \Gamma^2$, proving that the neuron's autaptic weight or reset will approach the optimal solution.

2 Learning in a homogeneous network

We now generalise our learning rule derivation to a network of N identical, homogeneously connected neurons. This generalisation is reasonably straightforward because many characteristics of the single neuron case are shared by a network of identical neurons. We will return to the more general case of heterogeneously connected neurons in the next section.

We begin by deriving optimal spiking dynamics, as in the single neuron case. This provides a target for learning, which we can then use to derive synaptic dynamics. As before, we want our network to produce spikes that optimally represent a variable x for a linear read-out. We assume that the read-out $\hat{x}$ is provided by summing and filtering the spike trains of all the neurons in the network:

$$\dot{\hat{x}} = -\hat{x} + \mathbf{\Gamma o}, \tag{12}$$

where the row vector $\mathbf{\Gamma} = (\Gamma, \ldots, \Gamma)$ contains the read-out weights[2] of the neurons and the column vector $\mathbf{o} = (o_1, \ldots, o_N)$ their spike trains. Here, we have used identical read-out weights for each neuron, because this indirectly leads to homogeneous connectivity, as we will demonstrate.

Next, we assume that a neuron only spikes if that spike reduces a loss-function. This spiking rule is similar to the single neuron spiking rule except that this time there is some ambiguity about which neuron should spike to represent a signal. Indeed, there are many different spike patterns that provide exactly the same estimate $\hat{x}$. For example, one neuron could fire regularly at a high rate (exactly like our previous single neuron example) while all others are silent. To avoid this firing rate ambiguity, we use a modified loss function, that selects amongst all equivalent solutions, those with the smallest neural firing rates. We do this by adding a 'metabolic cost' term to our loss function, so that high firing rates are penalised:

$$L = (x - \hat{x})^2 + \mu \|\bar{\mathbf{o}}\|^2, \tag{13}$$

where μ is a small positive constant that controls the cost-accuracy trade-off, akin to a regularisation parameter.

Each neuron in the optimal network will seek to reduce this loss function by firing a spike. Specifically, the i^{th} neuron will spike whenever $L(\text{no spike in } i) > L(\text{spike in } i)$. This leads to the following spiking rule for the i^{th} neuron:

$$V_i > T_i \tag{14}$$

where $V_i \equiv \Gamma(x - \hat{x}) - \mu\bar{o}_i$ and $T_i \equiv \Gamma^2/2 + \mu/2$. We can naturally interpret V_i as the membrane potential of the i^{th} neuron and T_i as the spiking threshold of that neuron. As before, we can now derive membrane potential dynamics:

$$\dot{\mathbf{V}} = -\mathbf{V} + \mathbf{\Gamma}^T c - (\mathbf{\Gamma}^T\mathbf{\Gamma} + \mu\boldsymbol{I})\mathbf{o}, \tag{15}$$

[2]The read-out weights must scale as $\Gamma \sim 1/N$ so that firing rates are not unrealistically small in large networks. We can see this by calculating the average firing rate $\sum_{i=1}^{N} \bar{o}_i/N \approx x/(\Gamma N) \sim \mathcal{O}(N/N) \sim \mathcal{O}(1)$.

where $\boldsymbol{I}$ is the identity matrix and $\boldsymbol{\Gamma}^T\boldsymbol{\Gamma} + \mu\boldsymbol{I}$ is the network connectivity. We can interpret the self-connection terms $\{\Gamma^2 + \mu\}$ as voltage resets that decrease the voltage of any neuron that spikes. This optimal network is equivalent to a network of identical integrate-and-fire neurons with homogeneous inhibitory connectivity.

The network has some interesting dynamical properties. The voltages of all the neurons are largely synchronous, all increasing to the spiking threshold at about the same time[3] (Fig. 1F). Nonetheless, neural spiking is asynchronous. The first neuron to spike will reset itself by $\Gamma^2 + \mu$, and it will inhibit all the other neurons in the network by Γ^2. This mechanism prevents neurons from spik-

[3]The first neuron to spike will be random if there is some membrane potential noise.

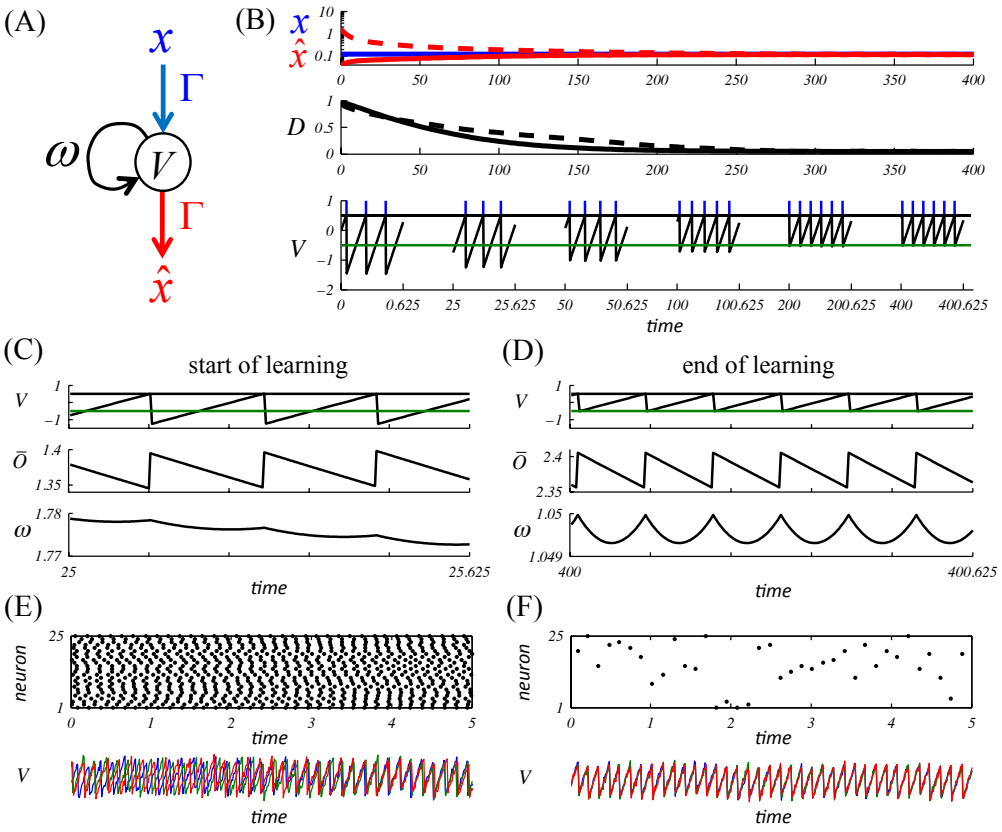

Figure 1: Learning in a single neuron and a homogeneous network. (A) A single neuron represents an input signal x by producing an output $\hat{x}$. (B) During learning, the single neuron output $\hat{x}$ (solid red line, top panel) converges towards the input x (blue). Similarly, for a homogeneous network the output $\hat{x}$ (dashed red line, top panel) converges towards x. Connectivity also converges towards optimal connectivity in both the single neuron case (solid black line, middle panel) and the homogeneous network case (dashed black line, middle panel), as quantified by $D = \max_{i,j}(\left|\Omega_{ij} - \Omega_{ij}^{opt}\right|^2 / \left|\Omega_{ij}^{opt}\right|^2)$ at each point in time. Consequently, the membrane potential reset (bottom panel) converges towards the optimal reset (green line, bottom panel). Spikes are indicated by blue vertical marks, and are produced when the membrane potential reaches threshold (bottom panel). Here, we have rescaled time, as indicated, for clarity. (C) Our learning rule dictates that the autapse ω in our single neuron (bottom panel) changes in proportion to the membrane potential (top panel) and the firing rate (middle panel). (D) At the end of learning, the reset ω fluctuates weakly about the optimal value. (E) For a homogeneous network, neurons spike regularly at the start of learning, as shown in this raster plot. Membrane potentials of different neurons are weakly correlated. (F) At the end of learning, spiking is very irregular and membrane potentials become more synchronous.

ing synchronously. The population as a whole acts similarly to the single neuron in our previous example. Each neuron fires regularly, even if a different neuron fires in every integration cycle.

The design of this optimal network requires the tuning of $N(N-1)$ synaptic parameters. How can an arbitrary network of integrate-and-fire neurons learn this optimum? As before, we address this question by using the optimal network as a target for learning. We start with an arbitrarily connected network of integrate-and-fire neurons:

$$\dot{\mathbf{V}} = -\mathbf{V} + \mathbf{\Gamma}^T c - \mathbf{\Omega}\mathbf{o}, \tag{16}$$

where $\mathbf{\Omega}$ is a matrix of connectivity weights, which includes the resets of the individual neurons. Assuming that learning occurs on a slow time scale, we can rewrite this equation as

$$\mathbf{V} = \mathbf{\Gamma}^T x - \mathbf{\Omega}\bar{\mathbf{o}}. \tag{17}$$

Now, repeating the arguments from the single neuron derivation, we modify the loss function to obtain an online learning rule. Specifically, we set $L_V = \|\mathbf{V}\|^2/2$, and calculate the gradient:

$$\frac{\partial L_V}{\partial \Omega_{ij}} = \sum_k V_k \frac{\partial V_k}{\partial \Omega_{ij}} = -\sum_k V_k \delta_{ki} \bar{o}_j - \sum_{kl} V_k \Omega_{kl} \frac{\partial \bar{o}_l}{\partial \Omega_{ij}}. \tag{18}$$

We can simplify this equation considerably by observing that the contribution of the second summation is largely averaged out under a wide variety of realistic conditions[4]. Therefore, it can be neglected, and we obtain the following local learning rule:

$$\tau \dot{\Omega}_{ij} = -\frac{\partial L_V}{\partial \Omega_{ij}} = V_i \bar{o}_j. \tag{19}$$

This is a Hebbian plasticity rule, whereby connectivity changes in proportion to the presynaptic firing rate $\bar{o}_j$ and post-synaptic membrane potential V_i. We assume that the neural thresholds are set to a constant T and that the neural resets are set to their optimal values $-T$. In the previous section we demonstrated that these resets can be obtained by a Hebbian plasticity rule (Eqn. (10)).

This learning rule minimises the difference between the read-out and the signal, by approaching the optimal recurrent connection strengths for the network (Fig. 1B). As in the single neuron case, learning does not stop, so the connection strengths fluctuate close to their optimal value. During learning, network activity becomes progressively more asynchronous as it progresses towards optimal connectivity (Fig. 1E, F).

3 Learning in the general case

Now that we have developed the fundamental concepts underlying our learning rule, we can derive a learning rule for the more general case of a network of N arbitrarily connected leaky integrate-and-fire neurons. Our goal is to understand how such networks can learn to optimally represent a J-dimensional signal $\mathbf{x} = (x_1, \ldots, x_J)$, using the read-out equation $\dot{\mathbf{x}} = -\mathbf{x} + \mathbf{\Gamma}\mathbf{o}$.

We consider a network with the following membrane potential dynamics:

$$\dot{\mathbf{V}} = -\mathbf{V} + \mathbf{\Gamma}^T \mathbf{c} - \mathbf{\Omega}\mathbf{o}, \tag{20}$$

where $\mathbf{c}$ is a J-dimensional input. We assume that this input is related to the signal according to $\mathbf{c} = \dot{\mathbf{x}} + \mathbf{x}$. This assumption can be relaxed by treating the input as the control for an arbitrary linear dynamical system, in which case the signal represented by the network is the output of such a computation [11]. However, this further generalisation is beyond the scope of this work.

As before, we need to identify the optimal recurrent connectivity so that we have a target for learning. Most generally, the optimal recurrent connectivity is $\mathbf{\Omega}^{\text{opt}} \equiv \mathbf{\Gamma}^T\mathbf{\Gamma} + \mu\boldsymbol{I}$. The output kernels of the individual neurons, $\mathbf{\Gamma}_i$, are given by the rows of $\mathbf{\Gamma}$, and their spiking thresholds by $T_i \equiv \|\mathbf{\Gamma}_i\|^2/2 +$

[4]From the definition of the membrane potential we can see that $V_k \sim \mathcal{O}(1/N)$ because $\Gamma \sim 1/N$. Therefore, the size of the first term in Eqn. (18) is $\sum_k V_k \delta_{ki} \bar{o}_j = V_i \bar{o}_j \sim \mathcal{O}(1/N)$. Therefore, the second term can be ignored if $\sum_{kl} V_k \Omega_{kl} \partial \bar{o}_l / \partial \Omega_{ij} \ll \mathcal{O}(1/N)$. This happens if $\Omega_{kl} \ll \mathcal{O}(1/N^2)$ as at the start of learning. It also happens towards the end of learning if the terms $\{\Omega_{kl} \partial \bar{o}_l / \partial \Omega_{ij}\}$ are weakly correlated with zero mean, or if the membrane potentials $\{V_i\}$ are weakly correlated with zero mean.

$\mu/2$. With these connections and thresholds, we find that a network of integrate-and-fire neurons will produce spike trains in such a way that the loss function $L = \|\mathbf{x} - \hat{\mathbf{x}}\|^2 + \mu\|\bar{\mathbf{o}}\|^2$ is minimised, where the read-out is given by $\hat{\mathbf{x}} = \mathbf{\Gamma}\bar{\mathbf{o}}$. We can show this by prescribing a greedy[5] spike rule: a spike is fired by neuron i whenever $L(\text{no spike in } i) > L(\text{spike in } i)$ [11]. The resulting spike generation rule is

$$V_i > T_i, \tag{21}$$

where $V_i \equiv \mathbf{\Gamma}_i^T(\mathbf{x} - \hat{\mathbf{x}}) - \mu\bar{o}_i$ is interpreted as the membrane potential.

[5]Despite being greedy, this spiking rule can generate firing rates that are practically identical to the optimal solutions: we checked this numerically in a large ensemble of networks with randomly chosen kernels.

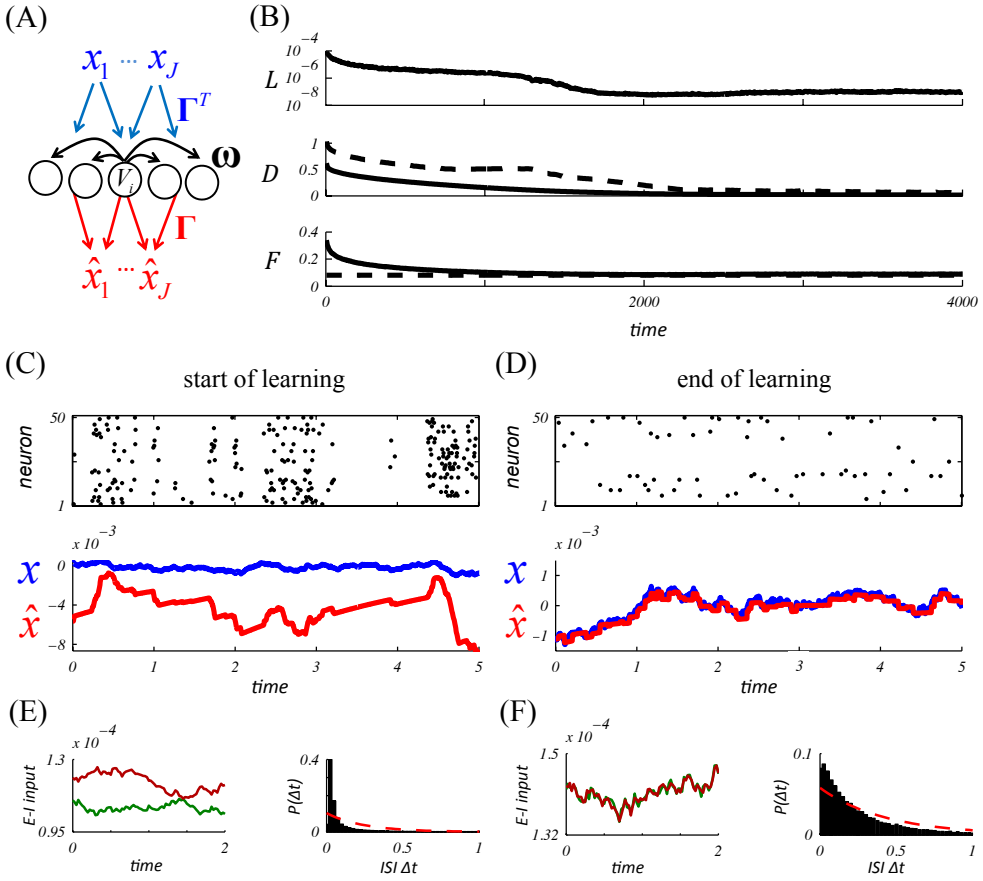

Figure 2: Learning in a heterogeneous network. (A) A network of neurons represents an input signal $\mathbf{x}$ by producing an output $\hat{\mathbf{x}}$. (B) During learning, the loss L decreases (top panel). The difference between the connection strengths and the optimal strengths also decreases (middle panel), as quantified by the mean difference (solid line), given by $D = \left\|\mathbf{\Omega} - \mathbf{\Omega}^{\text{opt}}\right\|^2 / \left\|\mathbf{\Omega}^{\text{opt}}\right\|^2$ and the maximum difference (dashed line), given by $\max_{i,j}(\left|\Omega_{ij} - \Omega_{ij}^{\text{opt}}\right|^2 / \left|\Omega_{ij}^{\text{opt}}\right|^2)$. The mean population firing rate (solid line, bottom panel) also converges towards the optimal firing rate (dashed line, bottom panel). (C, E) Before learning, a raster plot of population spiking shows that neurons produce bursts of spikes (upper panel). The network output $\hat{\mathbf{x}}$ (red line, middle panel) fails to represent $\mathbf{x}$ (blue line, middle panel). The excitatory input (red, bottom left panel) and inhibitory input (green, bottom left panel) to a randomly selected neuron is not tightly balanced. Furthermore, a histogram of interspike intervals shows that spiking activity is not Poisson, as indicated by the red line that represents a best-fit exponential distribution. (D, F) At the end of learning, spiking activity is irregular and Poisson-like, excitatory and inhibitory input is tightly balanced and $\hat{\mathbf{x}}$ matches $\mathbf{x}$.

How can we learn this optimal connection matrix? As before, we can derive a learning rule by minimising the cost function $L_V = \|\mathbf{V}\|^2/2$. This leads to a Hebbian learning rule with the same form as before:

$$\tau\dot{\Omega}_{ij} = V_i\bar{o}_j. \tag{22}$$

Again, we assume that the neural resets are given by $-T_i$. Furthermore, in order for this learning rule to work, we must assume that the network input explores all possible directions in the J-dimensional input space (since the kernels $\mathbf{\Gamma}_i$ can point in any of these directions). The learning performance does not critically depend on how the input variable space is sampled as long as the exploration is extensive. In our simulations, we randomly sample the input $\mathbf{c}$ from a Gaussian white noise distribution at every time step for the entire duration of the learning.

We find that this learning rule decreases the loss function L, thereby approaching optimal network connectivity and producing optimal firing rates for our linear decoder (Fig. 2B). In this example, we have chosen connectivity that is initially much too weak at the start of learning. Consequently, the initial network behaviour is similar to a collection of unconnected single neurons that ignore each other. Spike trains are not Poisson-like, firing rates are excessively large, excitatory and inhibitory input is unbalanced and the decoded variable $\hat{\mathbf{x}}$ is highly unreliable (Fig. 2C, E). As a result of learning, the network becomes tightly balanced and the spike trains become asynchronous, irregular and Poisson-like with much lower rates (Fig. 2D, F). However, despite this apparent variability, the population representation is extremely precise, only limited by the the metabolic cost and the discrete nature of a spike. This learnt representation is far more precise than a rate code with independent Poisson spike trains [11]. In particular, shuffling the spike trains in response to identical inputs drastically degrades this precision.

4 Conclusions and Discussion

In population coding, large trial-to-trial spike train variability is usually interpreted as noise [2]. We show here that a deterministic network of leaky integrate-and-fire neurons with a simple Hebbian plasticity rule can self-organise into a regime where information is represented far more precisely than in noisy rate codes, while appearing to have noisy Poisson-like spiking dynamics.

Our learning rule (Eqn. (22)) has the basic properties of STDP. Specifically, a presynaptic spike occurring immediately before a post-synaptic spike will potentiate a synapse, because membrane potentials are positive immediately before a postsynaptic spike. Furthermore, a presynaptic spike occurring immediately after a post-synaptic spike will depress a synapse, because membrane potentials are always negative immediately after a postsynaptic spike. This is similar in spirit to the STDP rule proposed in [12], but different to classical STDP, which depends on post-synaptic spike times [9].

This learning rule can also be understood as a mechanism for generating a tight balance between excitatory and inhibitory input. We can see this by observing that membrane potentials after learning can be interpreted as representation errors (projected onto the read-out kernels). Therefore, learning acts to minimise the magnitude of membrane potentials. Excitatory and inhibitory input must be balanced if membrane potentials are small, so we can equate balance with optimal information representation.

Previous work has shown that the balanced regime produces (quasi-)chaotic network dynamics, thereby accounting for much observed cortical spike train variability [13, 14, 4]. Moreover, the STDP rule has been known to produce a balanced regime [16, 17]. Additionally, recent theoretical studies have suggested that the balanced regime plays an integral role in network computation [15, 13]. In this work, we have connected these mechanisms and functions, to conclude that learning this balance is equivalent to the development of an optimal spike-based population code, and that this learning can be achieved using a simple Hebbian learning rule.

Acknowledgements

We are grateful for generous funding from the Emmy-Noether grant of the Deutsche Forschungsgemeinschaft (CKM) and the Chaire d'excellence of the Agence National de la Recherche (CKM, DB), as well as a James Mcdonnell Foundation Award (SD) and EU grants BACS FP6-IST-027140, BIND MECT-CT-20095-024831, and ERC FP7-PREDSPIKE (SD).

References

[1] Tolhurst D, Movshon J, and Dean A (1982) The statistical reliability of signals in single neurons in cat and monkey visual cortex. *Vision Res* **23**: 775–785.

[2] Shadlen MN, Newsome WT (1998) The variable discharge of cortical neurons: implications for connectivity, computation, and information coding. *J Neurosci* **18**(10): 3870–3896.

[3] Zohary E, Newsome WT (1994) Correlated neuronal discharge rate and its implication for psychophysical performance. *Nature* **370**: 140–143.

[4] Renart A, de la Rocha J, Bartho P, Hollender L, Parga N, Reyes A, & Harris, KD (2010) The asynchronous state in cortical circuits. *Science* **327**, 587–590.

[5] Ecker AS, Berens P, Keliris GA, Bethge M, Logothetis NK, Tolias AS (2010) Decorrelated neuronal firing in cortical microcircuits. *Science* **327**: 584–587.

[6] Okun M, Lampl I (2008) Instantaneous correlation of excitation and inhibition during ongoing and sensory-evoked activities. *Nat Neurosci* **11**, 535–537.

[7] Shu Y, Hasenstaub A, McCormick DA (2003) Turning on and off recurrent balanced cortical activity. *Nature* **423**, 288–293.

[8] Gentet LJ, Avermann M, Matyas F, Staiger JF, Petersen CCH (2010) Membrane potential dynamics of GABAergic neurons in the barrel cortex of behaving mice. *Neuron* **65**: 422–435.

[9] Caporale N, Dan Y (2008) Spike-timing-dependent plasticity: a Hebbian learning rule. *Annu Rev Neurosci* **31**: 25–46.

[10] Boerlin M, Deneve S (2011) Spike-based population coding and working memory. *PLoS Comput Biol* **7**, e1001080.

[11] Boerlin M, Machens CK, Deneve S (2012) Predictive coding of dynamic variables in balanced spiking networks. *under review*.

[12] Clopath C, Büsing L, Vasilaki E, Gerstner W (2010) Connectivity reflects coding: a model of voltage-based STDP with homeostasis. *Nat Neurosci* **13**(3): 344–352.

[13] van Vreeswijk C, Sompolinsky H (1998) Chaotic balanced state in a model of cortical circuits. *Neural Comput* **10**(6): 1321–1371.

[14] Brunel N (2000) Dynamics of sparsely connected networks of excitatory and inhibitory neurons. *J Comput Neurosci* **8**, 183–208.

[15] Vogels TP, Rajan K, Abbott LF (2005) Neural network dynamics. *Annu Rev Neurosci* **28**: 357–376.

[16] Vogels TP, Sprekeler H, Zenke F, Clopath C, Gerstner W. (2011) Inhibitory plasticity balances excitation and inhibition in sensory pathways and memory networks. *Science* **334**(6062):1569–73.

[17] Song S, Miller KD, Abbott LF (2000) Competitive Hebbian learning through spike-timing-dependent synaptic plasticity. *Nat Neurosci* **3**(9): 919–926.

Chapter 3

Learning an Auto-Encoder

In the previous article, we study how neurons in a population learn to communicate in order to efficiently and collectively represent the inputs they receive. The model consists of a recurrent network of leaky integrate-and-fire neurons that receive common inputs through random feedforward weights. The latter are fixed while the recurrent weights are trained to minimize the voltage fluctuations with a local plasticity rule. We show through simulations that the network converges to the optimal state as defined in (Boerlin and Denève, 2011; Boerlin et al., 2013). Such state is characterized by a tight balance between recurrent and feedforward inputs. Indeed, at the end of learning (the optimal state), the voltage is proportional to the error between the output and the input of the network. Thus if the two tightly match, the voltage fluctuations must also be small.

The Importance of Learning the Feedforward Weights

The previous study mainly focuses on recurrent weights. However, the distribution of the feedforward weight is also important to the quality of the representation. In this framework, the feedforward weight vector of each neuron defines the dimension along which the neuron represents the signals in the global output space. Indeed, badly distributed feedforward weight vectors may lead to the dismissal of important features in the input. To ensure that all the relevant dimensions are well represented, one can use large networks with a number of neurons much higher than the number of dimensions in the input. In this case, even randomly chosen feedforward weights will result in an even covering of the input space. In the previous study, we adopted this solution. Because the inputs we considered are low-dimensional (2 dimensions in figure 3), relatively small networks (50 neurons) will result in a dense coverage of the input space. However, the stimuli received by the brain such as images are very high-dimensional. In order to densely cover the space with random receptive fields, the use of a huge number of neurons is required. An alternative approach is to use few

neurons, but to choose their feedforward weights in an optimal way so as to capture the essential features in the input. The weights do not need to be set by hand, but can result from a learning procedure. More precisely, one can use a plasticity rule that adapts the feedforward weights to the input the neurons receive. In the following article we propose a solution to this problem and we derive a learning rule for the feedforward weights that enables the network to optimally represent its inputs.

Learning to Balance Should be Fast

The feedforward and recurrent weights are not learned separately and the respective learning rules work hand in hand. Assume that the network is already in a tightly balanced state thanks to the learning of the recurrent connections. Any change in the feedforward weights will likely cause an imbalance in the membrane potentials. In fact, the recurrent weights are no longer adapted to the new encoding weights and the recurrent input does not cancel the feedforward input as precisely as before. However, in order to converge, the learning of the feedforward weights requires the network to always be in the balanced state. For this reason, after each update of the encoding weights, recurrent connectivity should be relearned to always insure a tightly balanced state and thus an efficient encoding. The restoration of the balance should be very quick to ensure that the learning will converge in a reasonable time.

The learning rule that we derived in the previous work is indeed able to drive the network into the balanced state. However, it does not have the rapidity required to enable the learning of the feedforward weights simultaneously. A possible reason for this slow convergence is the presence of the firing rate term in the learning rule. This changes the weights continuously and on a timescale defined by the time constant of the decoder. However, and as shown in the supplementary information of the next article, the impact of the fast weights on balance and coding is defined on a much shorter timescale which is that of a single population's ISI. Thus, the presence of the rate term may introduce noisy and irrelevant updates that take time to be averaged out. We remedy this problem by deriving an equally local learning rule. However, this rule results from minimizing the membrane potential fluctuations only at spike times rather than continuously. The new learning rule is two orders of magnitude faster in converging to the balanced state (Figure 1E, Chapter 4). The network undergoing the feedforward and the recurrent learning rules indeed converges to the optimal state.

Towards More Biological Realism

In the examples considered so far, a neuron can be simultaneously excitatory and inhibitory. However, in biological networks, this is usually not the case. A neuron exclusively projects inhibitory or excitatory synapses. This principle is known as Dale's law. For this reason, we extend our approach to networks undergoing this constraint. In such networks, excitatory neurons inhibit each other with disynaptic inhibition. The Dale's and the non-Dale's network share the same learning rules.

We also extend the framework to the learning of non-whitened input. In all the previously considered examples, the input channels are independent. However, the signals received by the brain may be highly correlated such as nearby pixels in an image. Using a slight modification, we derived a feedfoward learning rule that is sensitive to the second-order statistics in the inputs. Indeed, the learning rule enables the network to optimally represent signals that contain correlations.

The paper is organized in two parts. The main manuscript explains the approach and the results without entering into the mathematical details. The Supplementary Information is devoted to the rigorous derivation of the network dynamics and the learning rules. We also present convergence proofs as well as the details and parameters of the simulations.

Learning to represent signals spike by spike

Wieland Brendel,[1,2,3,*] Ralph Bourdoukan,[2,*] Pietro Vertechi,[1,2,*]
Christian K. Machens,[1,†], Sophie Denève[2,†]

[1]Champalimaud Neuroscience Programme, Champalimaud Foundation, Lisbon, Portugal
[2]Group for Neural Theory, INSERM U960, Département d'Etudes Cognitives,
Ecole Normale Supérieure, Paris, France
[3]Werner Reichardt Centre for Integrative Neuroscience,
University of Tübingen, Germany
[*]These authors contributed equally
[†]To whom correspondence should be addressed;
E-mail: sophie.deneve@ens.fr or christian.machens@neuro.fchampalimaud.org

A key question in neuroscience is at which level functional meaning can be assigned to biophysical phenomena. The variability and redundancy of neural responses have often led to a dismissal of single neurons in favor of large populations. Here we show that many single neuron properties, such as variability of spiking, tuning to inputs, or balance of excitation and inhibition, emerge whenever a network learns to transmit information parsimoniously and precisely to the next layer. Using coding efficiency as an objective, we derive spike-timing-dependent learning rules for a recurrent neural network, and we provide exact solutions for the networks' convergence to an optimal state. As a result, we deduce an entire network from its input distribution and a firing cost. After learning, basic biophysical quantities such as voltages, firing thresholds, excitation, inhibition, or spikes acquire precise functional interpretations.

Many neural systems encode information by distributing it across the activities of large populations of spiking neurons. A lot of work has provided pivotal insights into the nature of the resulting population codes [1, 2, 3, 4, 5] and their generation through the internal dynamics of neural networks [6, 7, 8, 9]. However, we understand surprisingly little about the precise role of each individual spike in distributing information and in mediating learning.

We revisit this problem by studying a population of excitatory (E) neurons that are interconnected with inhibitory (I) interneurons (Fig. 1Ai). The excitatory neurons receive many input signals from other neurons within the brain. To encode these signals efficiently, each spike fired by an excitatory neuron should ideally contribute new and unique information to the population code. If each neuron receives a different input signal, this is easy. However, if two excitatory neurons receive similar inputs, they need to communicate with each other so as to not fire spikes for the same type of information. One possibility is that the inhibitory interneurons arbitrate such conflicts by creating competitive interactions between excitatory neurons [10]. How can neurons learn this from experience?

To formalize the problem, we will define a measure for the coding efficiency of a neural population (see Supplementary Information for mathematical details). First, we impose that any downstream area should be able to decode the input signals, $x_j(t)$, from a weighted sum of the neural responses, $\hat{x}_j(t) = \sum_{k=1}^{N} D_{jk} r_k(t)$, where D_{jk} is a decoding weight, and $r_k(t)$ is the postsynaptically filtered spike train of the k-th excitatory neuron. Second, we assume that the neurons fire as few spikes as possible, or, more generally, that they minimize a cost associated with firing, which we denote by $C(r)$. In other words, we measure the efficiency of the population code through an objective function that trades off accuracy for cost; this objective function is simply the sum of the coding error and the cost, $E = \sum_j (x_j - \hat{x}_j)^2 + C(r)$.

A single excitatory neuron has no access to this global objective function. Rather, it has access to the input signals, $x_j(t)$, that arrive via feedforward synapses, F_{ij}, and to the fil-

tered spike trains of other neurons, $r_k(t)$, that arrive via recurrent synapses, Ω_{ik}. For simplicity, we assume that neurons can be modeled as leaky integrate-and-fire neurons, and we will treat the inhibitory interneurons as simple relays for now (Fig. 1Aii). A neuron's membrane voltage is then the sum over all its integrated input currents, which here evaluates to $V_i(t) = \sum_{j=1}^{J} F_{ij} x_j(t) + \sum_{k=1}^{N} \Omega_{ik} r_k(t)$ (see Supplementary Information).

A crucial insight comes from studying a scenario in which the global coding error is quite small, so that $x_j(t) - \hat{x}_j(t) \approx 0$. If the recurrent connections are set to $\Omega_{ik} = -\sum_j F_{ij} D_{jk}$ (see Fig. 1Aii,iii), then the membrane potential of each neuron becomes $V_i(t) = \sum_j F_{ij}\big(x_j(t) - \hat{x}_j(t)\big) \approx 0$. Accordingly, the membrane potential now reflects a part of the global coding error, *despite* being computed from only feedforward and recurrent inputs. Each time this error becomes too large, the membrane potential reaches threshold. The neuron fires, updates the decoded input signal, and thereby decreases the error, as reflected in the voltage reset after a spike. Furthermore, through the recurrent synapses, the neuron will communicate the change in the global coding error to all neurons with similar feedforward inputs. In turn, any excitatory feedforward input into a neuron will immediately be counterbalanced by a recurrent inhibitory input (and vice versa). This latter reasoning links the precision of each neuron's code to the known condition of excitatory and inhibitory balance (EI balance) [11, 12, 13, 14, 15]. Indeed, balancing excitatory and inhibitory inputs optimally would minimize the variance of the membrane potential, and thus, the error projected in the direction of each neuron's feedforward weights [10, 16].

[Figure 1 about here]

How can a network of neurons learn to move into this very specific regime? Several learning rules for EI balance have been successfully proposed before [17, 18], and spike-timing-dependent plasticity (STDP) can even balance EI currents on a short time scale [18]. However,

here we both need to balance EI currents as precisely as possible, and we need to ensure convergence onto the right type of recurrent connectivity (Fig. 1Aii,iii). We will first examine the problem of EI balance more carefully by studying a neuron's membrane potential directly after it receives an inhibitory spike from one of its recurrent connections (Fig. 1Bi; black trace). After the inhibitory spike (Fig. 1Bi, red), the neuron integrates its feedforward input currents, leading to a transfer of electric charges across the membrane. The arrival of the second spike (Fig. 1Bi, blue) then causes a transfer of charges in the opposite direction, which ideally should cancel the total charge accumulated through the feedforward inputs. When the inhibitory spike overshoots (undershoots) its target, then the respective synapse needs to be weakened (strenghtened), see Fig 1Bii,iii. This scheme keeps the neuron's voltage (and thereby the coding error) perfectly in check. The regime can be reached by a simple voltage-based learning rule for the recurrent weights of neuron i, applied each time a presynaptic neuron k spikes,

$$\Delta\Omega_{ik} \propto -\beta(V_i + \mu r_i) - \Omega_{ik} - \mu\delta_{ik}. \tag{1}$$

Here V_i is the postsynaptic membrane potential before the arrival of the presynaptic spike, while β and μ are positive terms that implement a possible cost factor $C(r)$ (see Supplementary Information). In the absence of such costs ($\beta = 2$ and $\mu = 0$) the inhibitory weights will converge to the average postsynaptic membrane depolarization at the time of the presynaptic spike, scaled by a factor of two. As a result, the inhibitory spikes will systematically confine the membrane potential around its resting value.

Fig. 2 illustrates the effect of this learning rule in a network with 20 neurons receiving two random, time-varying inputs. Here the network was initialized with lopsided feed-forward weights and with recurrent weights equal to zero (Fig. 2Bi). While the network receives the random inputs, the recurrent synapses change according to the learning rule, Eq. 1, and each neuron thereby learns to balance its inputs. Once learnt, the recurrent connectivity reaches the desired

structure, $\Omega_{ik} = -\sum_j F_{ij} D_{jk}$ for some D_{jk}, and the voltages of the neurons become proportional to part of the coding error (see Supplementary Information for convergence proof). As a result of the EI balance, the voltages fluctuations of individual neurons are much better bounded around the resting potential (Fig. 2Eii), the global coding error decreases (Fig. 2Aii,Cii), and the network experiences a large drop in the overall firing rates (Fig. 2Aii,Dii).

Despite these overall improvements, however, the network still fails to represent part of the input, even after the recurrent connections have been learnt (Fig. 2Cii, arrow). Indeed, in the example provided, this part of the input signal cannot be properly represented because the feedforward connections do not cover the full two-dimensional signal space (Fig. 2Bii), which becomes particularly evident in the tuning curves of the individual neurons (Fig. 2Gii).

Consequently, the feedforward connections need to change as well, so that all parts of the input space are properly covered. We can again obtain a crucial insight by considering the final, 'learnt' state, in which case the feedforward connections are directly related to the optimal decoding weights, i.e., $F_{ik} = D_{ki}$ (see Supplementary Information). In Fig. 1C, we examine the decoding problem from the point of view of five neurons that seek to represent two input signals. If an input signal lies approximately in the direction of the vector of one of the neurons' decoding weights, then a few spikes suffice to represent it accurately (Fig. 1Ci). If the input signal lies elsewhere, many more spikes are required to achieve the same accuracy (Fig. 1Cii). For random input signals with zero mean and equal variance, as in Fig. 1Ci, the best representation is achieved when the decoding vectors are evenly distributed. (See Fig. 1D and Supplementary Information for correlated input signals.)

The feedforward weights of neuron j can learn to optimally cover the input space if they change each time neuron j fires a spike,

$$\Delta F_{ij} \propto \alpha x_i - F_{ij}, \tag{2}$$

where x_i is the feed-forward input signal, and $\alpha > 0$ is a scaling factor. In the case of correlated inputs, the term "F_{ij}" is replaced by the correlation of pre- and post-synaptic input currents (see Supplementary Information for details and convergence proofs).

From the perspective of standard frequency-modulated plasticity, the learning rule is Hebbian in that connections are reinforced for co-occurring high pre- and postsynaptic activity. Because of the competition introduced by the recurrent connections, a post-synaptic spike occurs only if no other neuron fired first in response to the same signal. This introduces repulsion between the feedforward weights of different neurons and eventually leads to an even coverage of the input space.

[Figure 2 about here]

The effect of the feedforward plasticity rule is shown in Fig. 2Aiii–Giii. The feedforward weights change until the input space is spanned more uniformly (Fig. 2Biii). While these changes are occuring, the recurrent weights remain plastic and keep the system in a balanced state. At the end of learning, the neuron's tuning curves are uniformally distributed (Fig. 2Giii), and the quality of the representation becomes optimal for all input signals (Fig. 2Aiii,Ciii).

Importantly, the final population code represents the input signals spike by spike. Initially, the neurons are unconnected (Fig. 2Bi), and their voltages reflect the smooth, time-varying input (Fig. 2Ei,Fi). Even with a bit of noise in the input currents, neurons fire the spikes at roughly the same time from trial to trial. After learning, the membrane potentials are correlated, reflecting their shared inputs, yet the individual spikes are far more susceptible to random fluctuations (Fig. 2Eiii,Fiii). Indeed, whichever neuron happens to fire first immediately inhibits (resets) the others, so that a small initial difference in the membrane potentials is sufficient to change the firing order completely. The random nature of spike timing is therefore a direct consequence of

a mechanism that prevents any redundant (or synchronous) firing. More generally, any source of noise or dependency on previous spike history will change the firing order, but without a significant impact on the precision of the code. Thus, variable spike trains co-exist with a highly reproducible and precise population code.

Fortunately, the same results can be obtained in networks with separate excitatory (E) and inhibitory (I) populations (Fig. 1Ai). In this more realistic case, the inhibitory population must simply learn to represent the population response of the excitatory population, after which it can balance the excitatory population in turn. This can be achieved if we train the EI connections using the feedforward rule (Eq. 2) while the II, EE, and IE connections are trained using the recurrent rule (Eq. 1; see Supplementary Information for details).

[Figure 3 about here]

Fig. 3 illustrates how the key results obtained in Fig. 2 hold in the the full EI network. The network converges to the optimal balanced state (Fig. 3B), and the precision of the representation improves substantially (Fig. 3Bi, Cii), despite the overall decrease in output firing rates (Fig. 3Bii, Cii). Initially regular and reproducible spike trains (Fig. 3Biii) become asynchronous, irregular, and comparable to independent Poisson processes (Fig. 3Biii, pairwise correlations are smaller than 0.001). Finally, we observe that the neuron's tuning curves, when measured along the first two signal directions, are bell-shaped just as in the previous example (Fig. 3Dii). Note that the inhibitory neurons fire more and have broader tuning than the excitatory neurons. This result holds independent of the chosen initial state of the network, and is simply owed to their smaller number.

[Figure 4 about here]

We have so far considered uncorrelated inputs. The case of correlated input signals is illustrated in Fig. 4 (see also supplementary Fig. S1). Here we trained a network to represent speech signals, filtered through 25 frequency channels, in its spiking output (Fig. 4A). Despite consisting of 100 neurons that fire at only ~ 4Hz the network learns to represent the signals with high precision (Fig. 4B,C). By learning the statistics of speech sounds (supplementary Fig. S2), the network becomes specialized for this type of signal. A new "non-speech" stimulus therefore results in poor EI balance, high firing rates, and poor coding (Fig. 4D,E). After experiencing the new sound several times, however, the network represents the "non-speech" sound as precisely and parsimoniously as the previously experienced speech sounds (Fig. 4F).

The accommodation of the network to the new stimulus is largely mediated by plasticity at the recurrent synapses, whereas the feedforward synapses are less essential. Indeed, turning off feedforward plasticity (but not recurrent plasticity) lets the network reach almost the same performance for the new stimulus, whereas turning off recurrent plasticity (but not feedforward plasticity) can even worsen the coding performance instead of improving it (supplementary Fig. S2).

Since learning the new stimulus relies on the recurrent connections re-balancing the feedforward inputs, EI balance should directly reflect behavioral performance, a prediction compatible with recent observations in the auditory cortex [19]. As a consequence, blocking inhibitory plasticity during perceptual learning should result in a worsening of EI balance and behavioral performance, while blocking excitatory plasticity should have more moderate effects.

In summary, we have shown how populations of excitatory and inhibitory neurons can learn to efficiently represent a signal spike by spike. This type of unsupervised learning, which includes both principal and independent component analysis as special cases [20], has previously been studied largely in rate networks [21, 22, 23, 24]. Implementations that seek to mimic biology by assuming spiking neurons, recurrent network architectures, or local learning rules have

been largely limited to heuristic or approximative approaches [25, 26, 27, 28]. Using a rigorous top-down approach, we have here derived biologically plausible rules that are guarantueed to converge to a specific connectivity and achieve a maximally efficient code. Besides solving the problem of unsupervised learning in spiking networks, our framework may provide a solid starting point to move to other types of computations. Thus, a second set of slower connections can implement arbitrary linear dynamics in our learnt networks [10], and the framework presented here may provide crucial intuitions for the learning of these connections as well.

Apart from these theoretical advances, many of the critical features that are hallmarks of cortical dynamics follow naturally from our framework, even though they were not included in the original objective. We list four of the most important features. First, the predicted spike trains are highly irregular and variable, which has indeed been widely reported in cortical neurons [29, 5]. However, this variability is a signature of the network's coding efficiency, rather than detrimental [14] or purposeful noise [30, 31]. Second, despite this spike train variability, the membrane potentials of similarly tuned neurons are strongly correlated (due to shared inputs), as has indeed been found in various sensory areas [32, 33]. Third, local and recurrent inhibition in our network serves to balance the excitatory feedforward inputs on a very fast time scale. Such EI balance, in which inhibitory currents track excitatory currents on a millisecond time scale has been found in various systems and under various conditions [34, 35]. Fourth, we have derived learning rules whose polarity depends on the relative timing of pre-and post-synaptic spikes (see insets in Fig. 3A). In fact, the respective sign switches simply reflect the immediate sign reversal of the coding error (and thus of the membrane potential) after each new spike. As a result, most connections display some features of the classic STDP rules, e.g. LTP for pre-post pairing, and LTD for post-pre pairing [36, 37]. The only exception are E-E connections that exhibit "reverse STDP", i.e. potentiation for post-pre pairing (Fig. 3A). Despite their simplicity, these rules are not only spike-time dependent but also weight and voltage-dependent, as

observed experimentally [26].

Our framework thereby bridges from the essential biophysical quantities, such as the membrane voltages of the neurons, to the resulting population code, while providing crucial new insights on learning and coding in spiking neural networks.

References and Notes

[1] A. P. Georgopoulos, A. B. Schwartz, R. E. Kettner, *Science* **233**, 1416 (1986).

[2] E. P. Simoncelli, B. A. Olshausen, *Annual review of neuroscience* **24**, 1193 (2001).

[3] E. Schneidman, M. J. Berry, R. Segev, W. Bialek, *Nature* **440**, 1007 (2006).

[4] B. B. Averbeck, P. E. Latham, A. Pouget, *Nature Reviews Neuroscience* **7**, 358 (2006).

[5] A. Wohrer, M. D. Humphries, C. K. Machens, *Progress in neurobiology* **103**, 156 (2013).

[6] S.-i. Amari, *Biological cybernetics* **27**, 77 (1977).

[7] R. Ben-Yishai, R. L. Bar-Or, H. Sompolinsky, *Proceedings of the National Academy of Sciences* **92**, 3844 (1995).

[8] C. Eliasmith, *Neural computation* **17**, 1276 (2005).

[9] Y. Burak, I. R. Fiete, *Proceedings of the National Academy of Sciences* **109**, 17645 (2012).

[10] M. Boerlin, C. K. Machens, S. Denève, *Plos Computiational Biology* **9**, e1003258 (2013).

[11] S. Denève, C. K. Machens, *Nature neuroscience* **19**, 375 (2016).

[12] C. van Vreeswijk, H. Sompolinsky, *Science* **274**, 1724 (1996).

[13] D. J. Amit, N. Brunel, *Cerebral cortex* **7**, 237 (1997).

[14] M. N. Shadlen, W. T. Newsome, *The Journal of neuroscience* **18**, 3870 (1998).

[15] A. Renart, *et al.*, *science* **327**, 587 (2010).

[16] M. Boerlin, S. Denève, *PLoS Comput Biol* **7**, e1001080 (2011).

[17] S. Song, K. D. Miller, L. F. Abbott, *Nature neuroscience* **3**, 919 (2000).

[18] T. Vogels, H. Sprekeler, F. Zenke, C. Clopath, W. Gerstner, *Science* **334**, 1569 (2011).

[19] B. J. Marlin, M. Mitre, J. A. D'amour, M. V. Chao, R. C. Froemke, *Nature* **520**, 499 (2015).

[20] A. Hyvärinen, J. Karhunen, E. Oja, *Independent component analysis*, vol. 46 (John Wiley & Sons, 2004).

[21] E. Oja, *Journal of mathematical biology* **15**, 267 (1982).

[22] A. J. Bell, T. J. Sejnowski, *Neural computation* **7**, 1129 (1995).

[23] P. Vertechi, W. Brendel, C. K. Machens, *Advances in Neural Information Processing Systems* (2014), pp. 3653–3661.

[24] C. Pehlevan, D. Chklovskii, *Advances in Neural Information Processing Systems* (2015), pp. 2269–2277.

[25] C. Savin, P. Joshi, J. Triesch, *PLoS Comput Biol* **6**, e1000757 (2010).

[26] C. Clopath, L. Büsing, E. Vasilaki, W. Gerstner, *Nature neuroscience* **13**, 344 (2010).

[27] J. Zylberberg, J. T. Murphy, M. R. DeWeese, *PLoS Comput Biol* **7**, e1002250 (2011).

[28] P. D. King, J. Zylberberg, M. R. DeWeese, *The Journal of Neuroscience* **33**, 5475 (2013).

[29] D. J. Tolhurst, J. A. Movshon, A. Dean, *Vision research* **23**, 775 (1983).

[30] J. Fiser, P. Berkes, G. Orbán, M. Lengyel, *Trends in cognitive sciences* **14**, 119 (2010).

[31] L. Buesing, J. Bill, B. Nessler, W. Maass, *PLoS Comput Biol* **7**, e1002211 (2011).

[32] J. F. Poulet, C. C. Petersen, *Nature* **454**, 881 (2008).

[33] J. Yu, D. Ferster, *Neuron* **68**, 1187 (2010).

[34] J. S. Isaacson, M. Scanziani, *Neuron* **72**, 231 (2011).

[35] M. Xue, B. V. Atallah, M. Scanziani, *Nature* **511**, 596 (2014).

[36] N. Caporale, Y. Dan, *Annu. Rev. Neurosci.* **31**, 25 (2008).

[37] D. E. Feldman, *Neuron* **75**, 556 (2012).

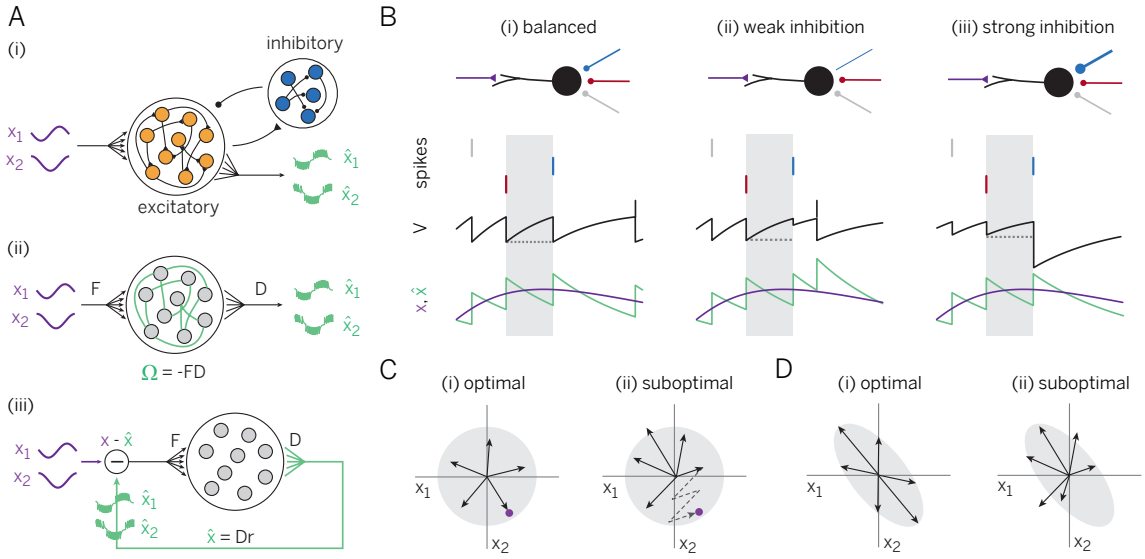

Figure 1: Networks learning to represent analog signals efficiently with spikes. **A.** (i) Recurrent neural network with input signal x (purple) and signal estimate $\hat{x}$ (green), as read out from the spike trains of the excitatory population. (ii) Simplified network without separate excitatory and inhibitory populations. (F=feedforward weights, D=decoding weights, Ω=recurrent weights) (iii) Same as (ii), but unfolded to illustrate the effect of the recurrent connections. **B.** A single neuron's EI balance as a target of learning for recurrent connections. (i) Ideal case with EI balance. (ii) One inhibitory synapse too weak. (iii) One inhibitory synapse too strong. Shown are the neuron's membrane voltage (black), spikes from three inhibitory neurons (vertical lines, color-coded by connection), signal (purple), and signal estimate (green). **C.** Influence of feedforward weights on signal encoding and decoding, shown for a five-neuron network encoding two signals with zero mean and equal variance (gray area). (i) Optimal scenario. (ii) Sub-optimal scenario. **D.** Similar to **C**, but for correlated input signals.

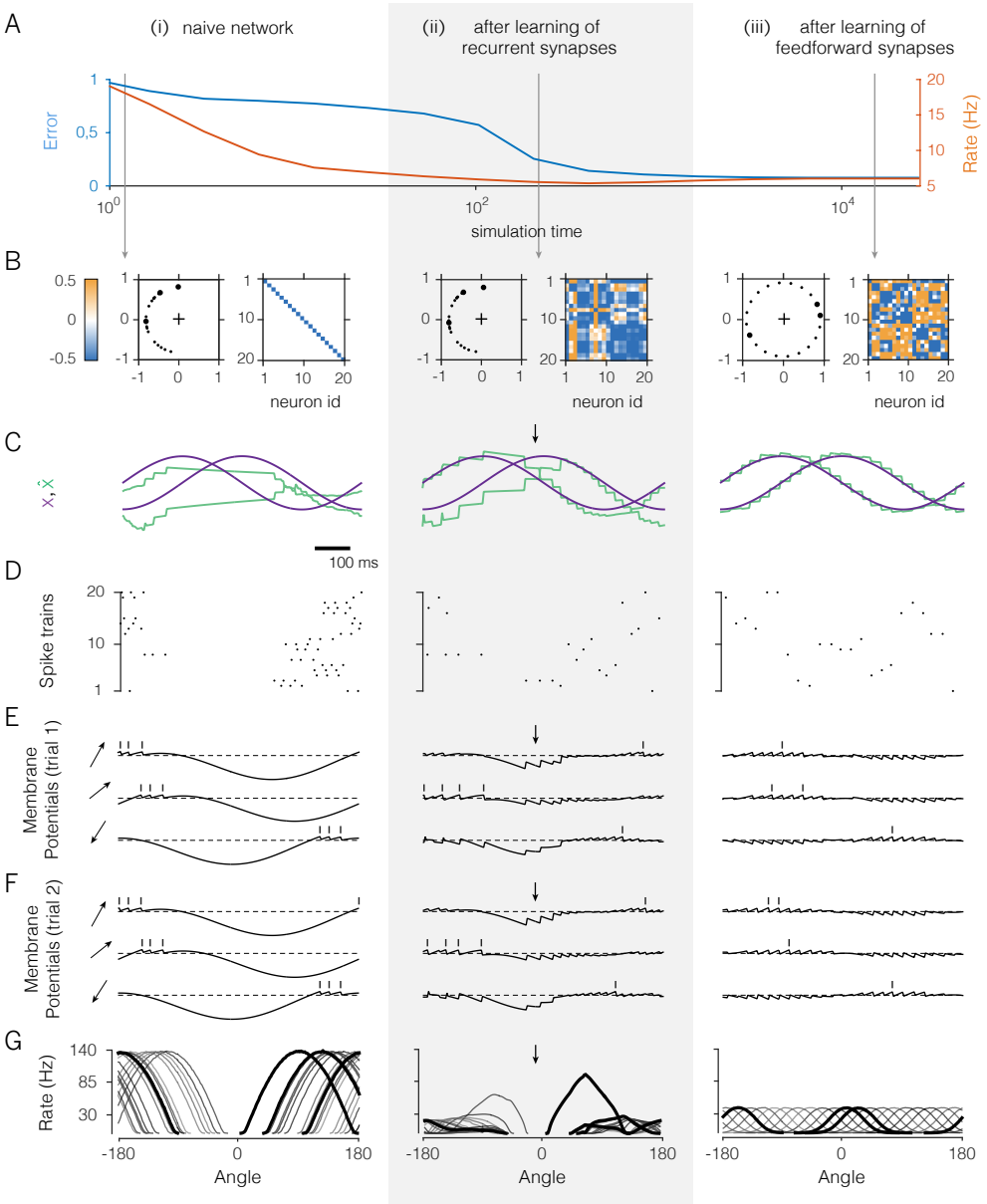

Figure 2: A 20-neuron network that learns to encode two signals. **A.** Evolution of coding error (blue) and mean population firing rate (orange) over learning. **B.** Feedforward and recurrent connectivity at three stages of learning. In each column, the left panel shows the two-dimensional feedforward weights, and the right panel the matrix of recurrent weights. Diagonal elements correspond to the neurons' self-resets after a spike. **C.** Example of time-varying input signals (purple) and signal estimates (green). Signal estimates in the naive network are constructed using an optimal linear decoder. Arrows indicate parts of the signal space that remain poorly represented, even after learning of the recurrent weights. **D.** Spike rasters from the network. **E.** Voltages and spike times of three exemplary neurons. Dashed lines illustrate the resting potential. **F.** As in **E**, but for a different trial. **G.** Tuning curves (firing rates as a function of an input with variable angle and constant radius in polar coordinates) of all neurons in the network.

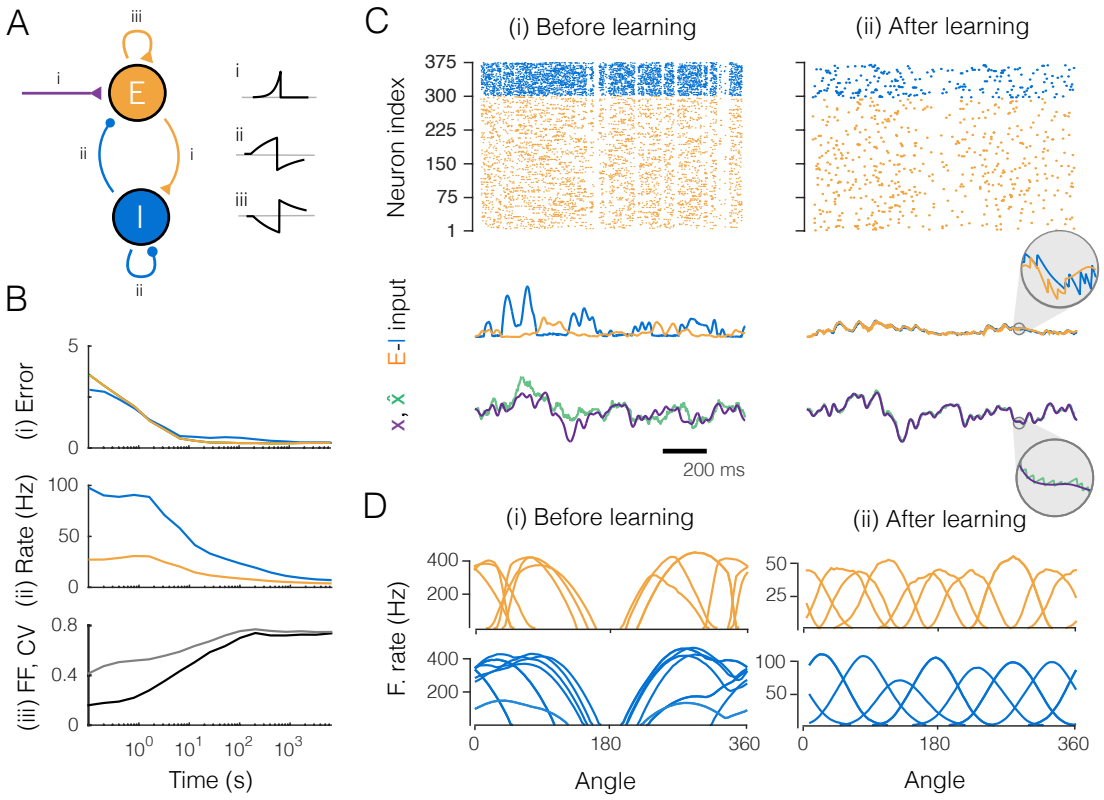

Figure 3: Large network (300 excitatory and 75 inhibitory neurons) that learns to encode three input signals. Excitation shown in orange, inhibition in blue. **A.** The EI network as in Fig. 1Ai and the learning rules (i, feedforward rule; ii and iii, recurrent rule). The insets show the results of an STDP-like protocol between pairs of neurons applied to our learning rules, with the x-axis representing the relative timing between pre- and post-synaptic spikes, and the y-axis the change in weight. **B.** Evolution of the network during learning. (i) Coding error for excitatory and inhibitory populations. (ii) Mean firing rate of excitatory and inhibitory populations. (iii) Averaged coefficient of variation (CV, gray) and Fano factor (FF, black) of the spike trains. **C.** Network input and output before (i) and after (ii) learning. (Top) Raster plots of spike trains from excitatory and inhibitory populations. (Center) Excitatory and inhibitory currents into one example neuron. After learning, inhibitory currents tightly balance excitatory currents (inset). (Bottom) One of the three input signals (purple) and the corresponding signal estimate (green) from the excitatory population. **D.** Tuning curves (firing rates as a function of the angle for two of the input signals, with the third signal clamped to zero) of the most active excitatory and inhibitory neurons.

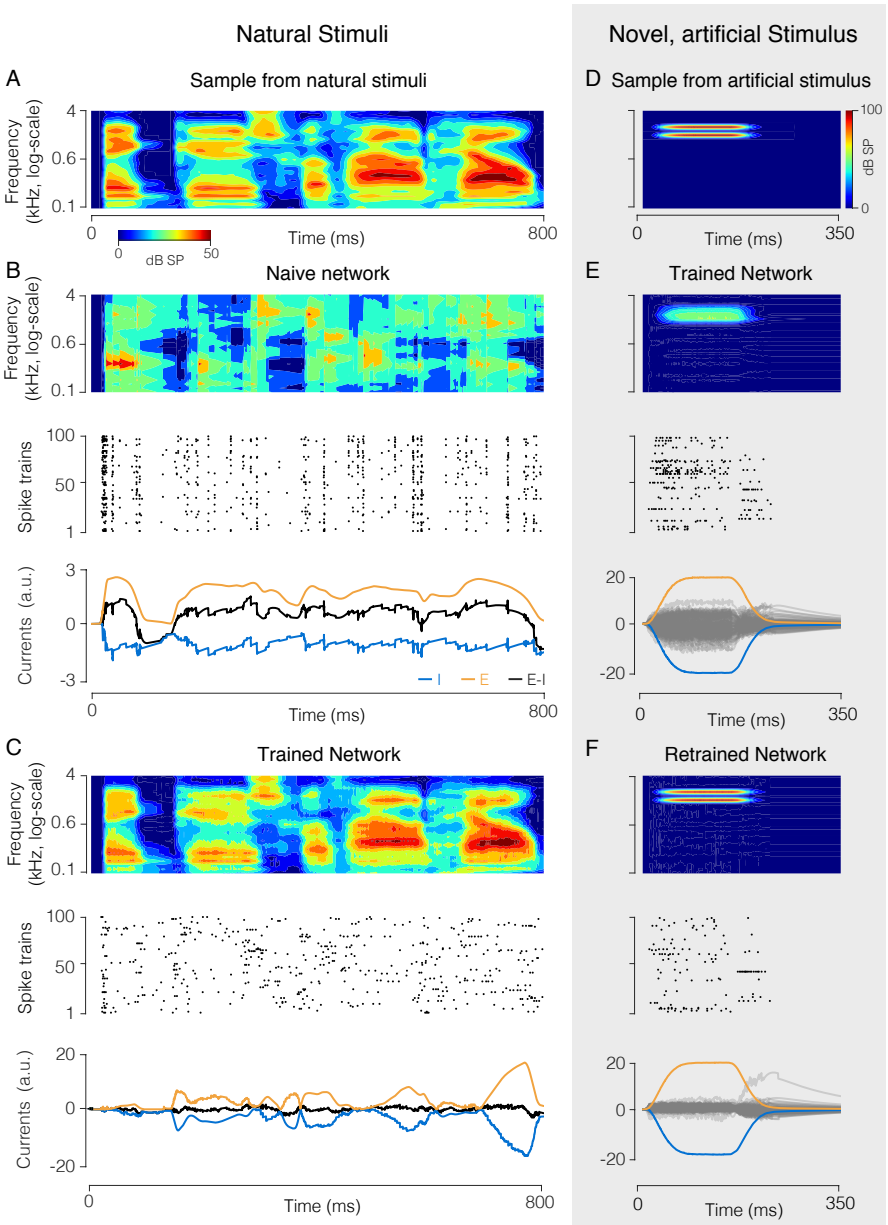

Figure 4: Network (100 neurons) that encodes a high-dimensional, structured natural input (speech sounds). **A.** Spectrogram of a speech sound. **B.** "Naive" network with random feedforward and recurrent weights. (Top) Optimal linear estimator applied to output spike trains reconstructs the stimulus poorly. (Center) Spike raster from all neurons, showing synchronous firing. (Bottom panel) Excitatory (orange) and inhibitory (blue) current into an example neuron are poorly balanced, causing large fluctuations in the total current (black). **C.** Same as **B**, after learning. The signal estimate tracks the signal closely (top), spike trains are asynchronous and irregular (center), and EI currents are tightly balanced (bottom). **D.** Spectrogram of artificial, "non-speech" sound. **E.** Response of the trained network trained to a non-speech sound, similar format as **B**, **C**. The new sound is improperly reconstructed (top), and EI responses are poorly balanced (bottom). Grey lines show the superposed total currents for all neurons, orange and blue lines show the mean excitatory and inhibitory currents, averaged over the population. **F.** Same as **E**, after re-training the network with a mixture of speech sounds and the new sound. The new sound is now represented precisely (top) with fewer spikes (center), and EI balance is improved (bottom).

Supplementary Material

Learning to represent signals spike by spike

Wieland Brendel*, Ralph Bourdoukan*, Pietro Vertechi*,
Christian K. Machens, and Sophie Denève

The supplementary material is structured as follows: In section 1 we describe the spiking recurrent neural networks that we are using throughout the paper, and we review the connectivity patterns of the networks that optimally encode the input signals in their spike trains. We then discuss the problem of learning the connectivity of the respective optimal networks in section 2 and provide the core intutions on learning in section 3, namely, that the tight balance between excitatory and inhibitory currents is the key signature of the optimal networks that needs to be learnt. In section 4 we then derive the voltage-based synaptic plasticity rules for the feedforward and recurrent connections that are used in the main paper, and we discuss their mathematical properties. In section 5 we apply these learning rules to the full EI network. All simulation parameters as well as pseudo-code are provided in Section 6, and additional simulations and details on the learning of the speech signals are presented in Section 7.

Contents

1 Spiking recurrent neural networks

In this section, we describe the network of integrate-and-fire neurons whose synaptic weights we will learn. We furthermore introduce an optimality criterion that we will use to define an 'optimal' spiking neural network, whose specific connectivity structure will provide the target of learning. We largely follow the derivations of [1, 2].

1.1 Inputs and outputs

We consider a recurrent neural network with N_E excitatory and N_I inhibitory neurons (compare Figure 1Ai in the main paper). The network receives a set of time-varying inputs $\mathbf{c}(t) = \big(c_1(t), c_2(t), \ldots, c_I(t)\big)$ and produces a set of spike train outputs from the excitatory population, $\mathbf{o}^E(t) = \big(o_1^E(t), o_2^E(t), \ldots, o_{N_E}^E(t)\big)$.[1] Each spike train is as a sum of Dirac delta functions, $o(t) = \sum_{t_k} \delta(t - t_k)$, where t_k are the spike times.

[1] In general, we will use bold-faced letters to indicate vectors or matrices, and italic letters to indicate scalar variables.

We furthermore define filtered versions of the input and output signals. First, the filtered input signal, $\mathbf{x}(t)$, is given by

$$\dot{\mathbf{x}}(t) = -\lambda \mathbf{x}(t) + \mathbf{c}(t), \tag{S.1}$$

and the filtered spike trains of the excitatory neurons are given by

$$\dot{\mathbf{r}}^E(t) = -\lambda \mathbf{r}^E(t) + \mathbf{o}^E(t). \tag{S.2}$$

Here, the parameter λ sets the decay rate of the respective variables. We note that for slowly changing signals, $\mathbf{x}(t)$ and $\mathbf{c}(t)$ are just scaled versions of each other. This is the scenario most applicable to our work, and we therefore refer to both variables as 'input signals'. Indeed, in the main text we did not distinguish between $\mathbf{c}(t)$ and $\mathbf{x}(t)$, but will do so here to be mathematically exact. We assume that these input signals are distributed according to some distribution $q(\mathbf{x})$. Throughout the first part of the supplementary information (SI), we will assume that this distribution is 'white', i.e., that its covariance matrix is the identity.

Each filtered spike trains can be viewed as a sum over postsynaptic potentials. We will sometimes refer to these filtered spike trains as 'instantaneous firing rates' or simply 'firing rates', even though, strictly speaking, the variables $r_i^E(t)$ have units of firing rates scaled by a factor $1/\lambda$. We note that this definition slightly deviates from [1]. The spike trains of the inhibitory interneurons are not considered to be part of the output. We will simply write $\mathbf{o}^I(t)$ for the vector of these spike trains, and $\mathbf{r}^I(t)$ for the respective filtered versions. Since there are N_I inhibitory neurons, both vectors are N_I-dimensional.

1.2 Voltage dynamics

We assume that the membrane voltages of both the excitatory and inhibitory neurons follow the dynamics of current-based, leaky integrate-and-fire neurons. Specifically, the voltage V_n^E of the n-th excitatory neuron is given by

$$\dot{V}_n^E(t) \equiv \frac{\partial V_n^E}{\partial t} = -\lambda V_n^E(t) + \mathbf{F}_n^E \cdot \mathbf{c}(t) + \mathbf{\Omega}_n^{EE} \cdot \mathbf{o}^E(t) + \mathbf{\Omega}_n^{EI} \cdot \mathbf{o}^I(t) + \sigma\eta(t), \tag{S.3}$$

where $\mathbf{F}_n^E$ are the feedforward weights of neuron n, $\mathbf{\Omega}_n^{EE}$ are the weights of the recurrent excitatory inputs, $\mathbf{\Omega}_n^{EI}$ are the weights of the recurrent inhibitory inputs, and $\sigma\eta(t)$ is a noise term. The multiplication sign '$\cdot$' denotes the inner product or dot product. We generally assume that the feedforward weights can be either excitatory or inhibitory, meaning that individual elements of $\mathbf{F}_n^E$ can be either positive or negative. The elements of $\mathbf{\Omega}_n^{EI}$ are assumed to be negative and the elements of $\mathbf{\Omega}_n^{EE}$ are assumed to be positive, with one exception: the self-connection weight Ω_{nn}^{EE} is assumed to be negative, as it determines the neuron's reset potential after a spike. Whenever the neuron hits a threshold, T_n^E, it fires a spike and resets its own voltage. In other words, we have included the reset of the integrate-and-fire neuron in its self-connection for mathematical convenience. After each spike, the voltage is therefore reset to $V_n^E \to T_n^E + \Omega_{nn}^{EE}$, where Ω_{nn}^{EE} is a negative number.

Similarly, the membrane voltage V_n^I of the n-th inhibitory neuron follows the dynamics

$$\dot{V}_n^I(t) \equiv \frac{\partial V_n^I}{\partial t} = -\lambda V_n^I(t) + \mathbf{\Omega}_n^{IE} \cdot \mathbf{o}^E(t) + \mathbf{\Omega}_n^{II} \cdot \mathbf{o}^I(t) + \sigma\eta(t), \tag{S.4}$$

where $\mathbf{\Omega}_n^{IE}$ are the recurrent weights from the excitatory population and $\mathbf{\Omega}_n^{II}$ are the recurrent weights from the inhibitory population. The thresholds are given by T_n^I and the reset is contained in the element Ω_{nn}^{II} of the recurrent inhibitory input.

1.3 Voltage dynamics without interneurons

To develop the learning rules for this network, it will prove quite useful to start with a simpler network in which we ignore the constraints imposed on biological networks due to Dale's law, i.e., due to the split of excitation and inhibition into separate pools of neurons. To this end, we will defer the treatment of the full EI network to section 5 and omit the inhibitory interneurons from now on by allowing direct inhibition between the excitatory neurons. We can then drop all EI subscripts and simplify the dynamics of the membrane voltage,

$$\dot{V}_n(t) = -\lambda V_n(t) + \mathbf{F}_n^\top \mathbf{c}(t) + \mathbf{\Omega}_n^\top \mathbf{o}(t) \tag{S.5}$$

where the neuron index n runs from $1 \dots N$ (with $N = N_E$) and the recurrent weights $\mathbf{\Omega}_n \in \mathbb{R}^N$ can be both excitatory or inhibitory. We also left out the noise term, since it will essentially be irrelevant for the derivation of the learning rules. The treatment of the full EI network can be found in section 5. For notational convenience, we will furthermore write all inner products as matrix products, so that $\mathbf{F}_n \cdot \mathbf{c}(t) = \mathbf{F}_n^\top \mathbf{c}(t)$ etc.

The synaptic weights of the individual neurons can be combined to yield the connectivity matrices of the network. Throughout the SI, we define the $N \times I$ matrix of feedforward weights as $\mathbf{F} = [\mathbf{F}_1, \mathbf{F}_2, \dots, \mathbf{F}_N]^\top$ and the $N \times N$ matrix of recurrent weights as $\mathbf{\Omega} = [\mathbf{\Omega}_1, \dots, \mathbf{\Omega}_N]^\top$. This allows us to write the dynamics of the whole network in the compact form

$$\dot{\mathbf{V}}(t) = -\lambda \mathbf{V}(t) + \mathbf{F}\mathbf{c}(t) + \mathbf{\Omega}\mathbf{o}(t),$$

where $\mathbf{V}$ is simply the N-dimensional vector of all voltages. Finally, we can formally integrate the differential equation, using (S.1) and (S.2), to obtain

$$\mathbf{V}(t) = \mathbf{F}\mathbf{x}(t) + \mathbf{\Omega}\mathbf{r}(t). \tag{S.6}$$

Note that this integration does not constitute an explicit solution to the differential equation, since the instantaneous firing rates, $\mathbf{r}(t)$, appear on the r.h.s. However, the integration highlights the particular relation between the voltages and the filtered spike trains, which will become useful further below.

1.4 Readouts and objectives

The network receives the time-varying input signals, $\mathbf{x}(t)$, and generates a set of output spike trains, $\mathbf{o}(t)$. We will assume that a downstream area will seek to construct an estimate $\hat{\mathbf{x}}(t)$ of the input signal from a weighted sum of the

filtered output spike trains, [2]

$$\begin{aligned}\hat{\mathbf{x}} &= \mathbf{D}\mathbf{r} \\ &= \sum_{n=1}^{N_E} \mathbf{D}_n r_n\end{aligned} \tag{S.7}$$

where $\mathbf{D}$ is an $I \times N_E$ matrix of decoder weights, and $\mathbf{D}_n$ are the columns of this matrix. The vector $\mathbf{D}_n$ summarizes the contribution of neuron n to the reconstruction of the signal.[3] For future reference, we note that we can use (S.2) to obtain a differential equation for this readout,

$$\dot{\hat{\mathbf{x}}} = -\lambda\hat{\mathbf{x}} + \mathbf{D}\mathbf{o}.$$

We can measure the quality of any readout by averaging its performance over a time interval T. To denote time averages of a quantity $z(t)$, we will use angular brackets so that $\langle z(t)\rangle_t = \frac{1}{T}\int_0^T z(t)$. With this in mind, we define the following loss function,

$$\begin{aligned}L &= \langle \ell(t)\rangle_t \\ &\equiv \left\langle \|\mathbf{x}(t) - \hat{\mathbf{x}}(t)\|^2 + C\big(\mathbf{r}(t)\big)\right\rangle_t\end{aligned}$$

where the first term inside the brackets is a quadratic measure of the reconstruction error, the second term is a cost term on the firing rates. If the time interval T over which this loss is averaged is large compared to the time-scale of the input, then the distribution of inputs $\mathbf{c}$ during this interval is approximately the same as the true input distribution $q(\mathbf{c})$. Hence, for large T we essentially sample the loss evenly over the distribution $q(\mathbf{c})$ of all possible inputs $\mathbf{c}$ (and hence over the distribution of input signals $q(\mathbf{x})$, since the signals $\mathbf{x}$ are filtered versions of $\mathbf{c}$). To denote expectation values of a quantity $z(\mathbf{x})$ with respect to the distribution $q(\mathbf{x})$, we will again use angular brackets, writing $\langle z(\mathbf{x})\rangle_{q(\mathbf{x})} = \int d\mathbf{x} q(\mathbf{x}) z(\mathbf{x})$. Accordingly, we can rewrite (S.8) as an estimate of the expected loss over the inputs,[4]

$$\begin{aligned}L &= \langle \ell(\mathbf{x})\rangle_{q(\mathbf{x})} \\ &= \left\langle \|\mathbf{x} - \hat{\mathbf{x}}\|^2 + C(\mathbf{r})\right\rangle_{q(\mathbf{x})}.\end{aligned} \tag{S.8}$$

For ease of notation we will typically suppress the difference between these two formulations and simply use angular brackets, $\langle z\rangle$, to denote averaging of the variable z over either time or input signals. Similarly, we will write

$$\ell = \|\mathbf{x} - \hat{\mathbf{x}}\|^2 + C(\mathbf{r}) \tag{S.9}$$

[2] Please note that we will generally drop the explicit notion of time-dependence to streamline the presentation, and so we here write $\hat{\mathbf{x}}$ instead of $\hat{\mathbf{x}}(t)$ and $\mathbf{r}$ instead of $\mathbf{r}(t)$.

[3] Please note that $\mathbf{D}_n$ denotes a *column* of the decoder matrix $\mathbf{D}$ whereas $\mathbf{F}_n$, for instance, denotes a *row* of the matrix of excitatory feedforward weights. Nonetheless, all vectors, including these two, are assumed to be column vectors.

[4] We previously assumed that the distribution $q(\mathbf{x})$ is white, i.e., its covariance matrix is the identity. For non-white signals, it is advantageous to modify the definition of the loss, and we will discuss this more general case in section 4.8.

to refer either to the time-dependent loss $\ell(t)$ or the signal-dependent loss $\ell(\mathbf{x})$.

The cost term, $C(\mathbf{r})$, allows us to assign a 'cost' to the representation (in terms of filtered spike trains) chosen by a particular network. Typical choices for the cost-term are $C(\mathbf{r}) = \sum_i r_i^2 = \|\mathbf{r}\|^2$ or $C(\mathbf{r}) = \sum_i |r_i| = \|\mathbf{r}\|_1$. (We will discuss the rationale for these choices in section 1.8.) For concreteness, we adopt a linear sum of the two, but many results and intuitions directly generalize to other cost functions. The objective (S.9) then reads

$$\ell = \|\mathbf{x} - \hat{\mathbf{x}}\|^2 + \mu \|\mathbf{r}\|^2 + \nu \|\mathbf{r}\|_1 . \tag{S.10}$$

1.5 The optimal decoder

The cost function not only allows us to define the quality of a particular reconstruction, $\hat{\mathbf{x}} = \mathbf{D}\mathbf{r}$, but it also allows us to determine the best possible decoder $\mathbf{D}$ for a given network. The corresponding optimization problem is identical to linear regression. Taking the derivative of (S.8) with respect to $\mathbf{D}$, we have

$$\frac{\partial L}{\partial \mathbf{D}} = -2 \left\langle (\mathbf{x} - \hat{\mathbf{x}})\mathbf{r}^\top \right\rangle ,$$

which we can set to zero to obtain the well-known linear regression solution,

$$\mathbf{D} = \left\langle \mathbf{x}\mathbf{r}^\top \right\rangle \left\langle \mathbf{r}\mathbf{r}^\top \right\rangle^{-1} . \tag{S.11}$$

For each particular network architecture with feedforward weights $\mathbf{F}$ and recurrent weights $\mathbf{\Omega}$, we can employ this formula to find the respective optimal decoder. Indeed, we used this formula in Figures 2–4 in the main text, whenever we reconstruct the input signal from the spike trains of a naive, unlearnt network.

1.6 The optimal decoder under length constraints

The (optimal) decoder will be an important conceptual quantity for the learning rules of the spiking network. In that context, however, the optimal decoder will be constrained to be of a particular length. While the necessity of this constraint will only become clear below, we here describe the corresponding optimization problem for future reference. (This subsection can also be skipped on first reading.)

Specifically, we will constrain the neurons' individual contributions to the readout, $\mathbf{D}_n$, to be of a certain length. Using a set of Lagrangian multipliers λ_n, we simply add these length constraints to the mean square error of the reconstruction error, (S.8), to obtain the modified loss function [5]

$$L = \left\langle \|\mathbf{x} - \hat{\mathbf{x}}\|^2 \right\rangle + \sum_{n=1}^{N_E} \lambda_n \left(\|\mathbf{D}_n\|^2 - a_n \right) ,$$

where a_n specifies the length of the n-th decoder $\mathbf{D}_n$. Taking the derivative of the loss with respect to $\mathbf{D}_n$ (and remembering that $\hat{\mathbf{x}} = \sum_n \mathbf{D}_n r_n$) yields

$$\frac{\partial L}{\partial \mathbf{D}_n} = -2 \langle (\mathbf{x} - \hat{\mathbf{x}}) r_n \rangle + 2\lambda_n \mathbf{D}_n .$$

[5] Note that we leave out the cost term $C(\mathbf{r})$ for simplicity, since it does not depend on the decoder.

In turn, we can set the derivative to zero to find the minimum of the loss function. An insightful *implicit* solution is found by writing

$$\mathbf{D}_n = \frac{1}{\lambda_n} \langle (\mathbf{x} - \hat{\mathbf{x}}) r_n \rangle \,, \tag{S.12}$$

and illustrates that the decoder will generally align with the direction of the largest reconstruction errors whenever the neuron fires strongly. An explicit solution can be found as well, corresponding to a penalized least square solution, and is given by

$$\mathbf{D} = \left\langle \mathbf{x}\mathbf{r}^\top \right\rangle \left\langle \mathbf{r}\mathbf{r}^\top + \mathbf{\Lambda} \right\rangle^{-1} .$$

where $\mathbf{\Lambda}$ is a diagonal matrix whose entries are the Lagrangian constraints, λ_n. This equation can then be compared with the unconstrained solution, (S.11).

1.7 The connectivity and thresholds of the optimal network

The optimal decoder allows us to find the best possible reconstruction for a given network. We emphasize that the 'best possible reconstruction' is not necessarily a good one; depending on the network connectivity, the reconstruction may not work at all. To counter this problem, we will study how to choose the free internal parameters of the network—the synaptic weights and the thresholds—such that the loss function is minimized. In other words, instead of assuming that the network is given, and then optimizing the decoder $\mathbf{D}$, we will now assume that the decoder is given, and then optimize the network. We will do so following the derivations layed out in [1].

First, we assume that we have a given and fixed decoder, $\mathbf{D}$. We can then consider the effect of a spike of neuron n on the loss ℓ (S.10). The spike will increase the filtered spike train, $r_n \to r_n + 1$, and thus update the signal estimate according to $\hat{\mathbf{x}} \to \hat{\mathbf{x}} + \mathbf{D}_n$, where $\mathbf{D}_n$ is the n-th column of the decoder matrix $\mathbf{D}$. We can therefore rewrite the objective (S.10) after the firing of the spike as

$$\ell(\text{neuron } n \text{ spiked}) = \|\mathbf{x} - \hat{\mathbf{x}} - \mathbf{D}_n\|^2 + \mu \|\mathbf{r} + \mathbf{e}_n\|^2 + \nu \|\mathbf{r} + \mathbf{e}_n\|_1 \,,$$

where $[\mathbf{e}_n]_j = \delta_{nj}$. We adopt a greedy optimization scheme in which a neuron fires as soon as its spike decreases the loss. Mathematically, this condition yields the expression $\ell(n \text{ spiked}) < \ell(n \text{ did not spike})$, from which we can immediately derive the spiking condition [2, 1],

$$\mathbf{D}_n^\top \mathbf{x} - \mathbf{D}_n^\top \mathbf{D}\mathbf{r} - \mu r_n > \frac{1}{2}(\|\mathbf{D}_n\|_2^2 + \mu + \nu).$$

Notice that all terms on the l.h.s. are time-dependent while all terms on the r.h.s. are constant. We will now show that we can identify the l.h.s. of this inequality with the (time-varying) voltages of our neurons, and the r.h.s. with the (constant) thresholds,

$$V_n(t) = \mathbf{D}_n^\top \mathbf{x} - \mathbf{D}_n^\top \mathbf{D}\mathbf{r} - \mu r_n, \tag{S.13}$$

$$T_n = \frac{1}{2}(\|\mathbf{D}_n\|^2 + \mu + \nu). \tag{S.14}$$

To see why we can make this equivalence, the reader can either compare the voltage in the above equation with (S.6), or we can take the temporal derivative of V_n to obtain

$$\begin{aligned}\dot{V}_n &= \mathbf{D}_n^\top \dot{\mathbf{x}} - \mathbf{D}_n^\top \mathbf{D}\dot{\mathbf{r}} - \mu \dot{r}_n, \\ &= \mathbf{D}_n^\top(-\lambda \mathbf{x} + \mathbf{c}) - \mathbf{D}_n^\top \mathbf{D}(-\lambda \mathbf{r} + \mathbf{o}) - \mu(-\lambda r_n + o_n), \\ &= -\lambda V_n + \mathbf{D}_n^\top \mathbf{c} - (\mathbf{D}^\top \mathbf{D}_n + \mu \mathbf{e}_n)^\top \mathbf{o},\end{aligned}$$

where we used (S.1) and (S.2) in the second line, and (S.13) in the third line.

This differential equation for the membrane voltage can now be compared with the equation for the general network of integrate-and-fire neurons, (S.5). To be optimal, a network should therefore have the following connectivity,

$$\begin{aligned}\mathbf{F} &= \mathbf{D}^\top, \\ \mathbf{\Omega} &= -\mathbf{D}^\top \mathbf{D} - \mu \mathbf{I} \\ &= -\mathbf{F}\mathbf{F}^\top - \mu \mathbf{I},\end{aligned}$$

where $\mathbf{I}_N$ is the $N \times N$ identity matrix.[6] These connectivities are similar to those of optimal rate networks that are designed to minimize the loss function (S.10) [3, 4, 5, 6]. They are quite specific in that the recurrent weights are symmetric and, as shown in the last equation, are directly related to the feedforward weights. Furthermore, the thresholds are similarly related to the recurrent (and thereby the feedforward connectivities) via (S.14) so that

$$\begin{aligned}\mathbf{T} &= \frac{1}{2}\left(\mathrm{diag}(\mathbf{D}^\top \mathbf{D}) + \mu + \nu\right) \\ &= \frac{1}{2}\left(-\mathrm{diag}(\mathbf{\Omega}) + \nu\right),\end{aligned}$$

where $\mathbf{T}$ is simply the N-dimensional vector of all thresholds.

An important observation is that the spiking condition of a single neuron relies on local information only and does not require the evaluation or knowledge of the full objective function. Indeed, rewriting (S.13) by using the definition of $\hat{\mathbf{x}}$, we obtain

$$V_n(t) = \mathbf{D}_n^\top(\mathbf{x} - \hat{\mathbf{x}}) - \mu r_n, \tag{S.15}$$

so that the voltage reflects both the part of the reconstruction error, $\mathbf{x} - \hat{\mathbf{x}}$, that is projected onto the decoder weights $\mathbf{D}_n$, as well as the quadratic cost term. Accordingly, the voltage acquires a precise functional interpretation in the optimal network. Furthermore, a spike fired by a neuron is designed to decrease this projected error, and in turn decreases the objective [1].

1.8 The cost-term revisited

With the structure of the optimal network in mind, we can reconsider the importance of the cost term, $C(\mathbf{r})$. To do so, we imagine that there are (many)

[6] Note that, since we are now ignoring Dale's law, all synaptic weights can be both excitatory and inhibitory. For pedagogical purposes, however, it is often useful to consider the case in which all feedforward weights are excitatory, and hence all lateral weights are inhibitory (in the optimal network). We made use of this scenario in Figure 1 of the main paper, and will use it in the SI, as well

more neurons than independent inputs, i.e., $N > I$. Ignoring spikes for a moment, and assuming graded firing rates $\mathbf{r}$ and constant inputs $\mathbf{x}$, we observe that the solution space of (S.8) is degenerate in that many possible combinations of firing rates can minimize the objective. Possible solutions are those in which only a few neurons fire with high rates (sparse regime), and those in which many neurons fire with fairly low firing rates (dense regime).

The cost term allows us to control which solution the network converges to. The typical choice to enforce sparse population responses is a so-called *L1-cost*, $C(\mathbf{r}) = \|\mathbf{r}\|_1 = \sum_n |r_n|$, and the typical choice to enforce a dense population code is a so-called *L2-cost* $C(\mathbf{r}) = \|\mathbf{r}\|_2^2 = \sum_n r_n^2$. Many other costs such as slowness, group sparsity and others have been extensively discussed in the literature, especially in the context of regularization. We here adopt a linear sum of L1- and L2-cost for the rest of this manuscript, but the generalisation to other cost functions is possible with typically few modifications (see also [1] for a more detailed discussion).

2 The learning problem

Learning in recurrent neural networks can have different connotations. To clarify why we consider it a difficult problem, we will first review classical approaches and then describe exactly which problem we are trying to tackle.

2.1 Classical approaches to learning in recurrent neural networks

In *'top-down'* approaches, one first specifies an objective that quantifies a network's performance in a particular task, and then derives learning rules that modify the connectivity of the network to improve task performance. While this approach has mostly been used in *feedforward* neural networks, several approaches exist for *recurrent* neural networks [8]. Famous examples include the Hebbian rules for Hopfield networks [7], 'backpropagation in time' [9], or the more recent 'FORCE' learning algorithm [10].

The key problem with top-down approaches is that they often achieve their functionality at the cost of biological plausibility. First, almost all top-down approaches are based on 'firing rate' networks, i.e., networks in which neurons communicate with continuous rates rather than spikes. Second, most top-down approaches lead to 'learning rules' that are non-local, i.e., that depend on non-local information, such as the activity or synaptic weights of other neurons in the network.[7]

In *'bottom-up'* approaches, the word 'learning' refers to networks whose synapses undergo specific, biologically plausible plasticity rules. In contrast to top-down approaches, the bottom-up perspective has allowed researchers to explicitly investigate the effect of spike-timing-dependent plasticity on the dynamics of spiking networks. For instance, such rules may allow a network to learn how to properly balance itself on a specified time scale [11, 14].

The key problem with bottom-up approaches is that the generation of a particular dynamical regime does not necessarily imply any specific functionality. Indeed, bottom-up approaches usually run into problems when trying to

[7] The Hebbian rule for Hopfield networks is a famous exception to this case.

tie synaptic plasticity to computational goals. While several studies have delineated rules of thumb for how plausible learning rules may give rise to certain functions, e.g. [15], they lack the insight or the mathematical guarantees that come with top-down approaches.

The difficulty of learning in recurrent neural networks is manifest in the gap between these two approaches: top-down approaches achieve functionality by sacrificing biological realism, and bottom-up approaches achieve biological realism by sacrificing functionality. A crucial challenge in bridging this gap turns out to be the locality constraint: biological synapses can only rely on local information in order to modify their strength. Indeed, deriving learning rules from an objective function under this locality constraint is often analytically intractable, and a large literature is devoted to this problem (distributed optimization; game theory; etc.). Most attempts at deriving local learning rules from functional principles have tried to circumvent this issue by approximating derived learning rules with local ones (e.g. [2, 5]). However, these approaches tend to work only under certain conditions and carry few mathematical guarantees in terms of optimality or convergence [16].

2.2 The efficient coding objective

Here our goal will be to bridge the gap and derive biologically realistic, local learning rules for a spiking network, that are guarantueed to converge to a specific global optimum. We will do this for the 'autoencoder', an unsupervised learning system that receives a set of time-varying input signals, and then learns to generate an efficient spike code in order to represent these signals. Given that our work is based on minimizing quadratic loss functions with linear and quadratic costs, algorithms such as principal component analysis (PCA) and independent component analysis (ICA) are special cases of this class of learning problems [17].

We will start by defining the objective of our network. In Section 1.7, we assumed a *given and fixed* decoder $\mathbf{D}$, and then derived a network whose spike times $\mathbf{o}(t)$ minimize the objective [8]

$$L^* = \min_{\mathbf{o}} \left\langle \|\mathbf{x} - \mathbf{D}\mathbf{r}\|^2 + \mu \|\mathbf{r}\|^2 + \nu \|\mathbf{r}\|_1 \right\rangle .$$

Unfortunately, this objective cannot be used for learning a network, since the pre-defined decoder $\mathbf{D}$ is not part of the network structure, and there is a priori no way the network can guess it.[9] Indeed, learning has to start in a network with random initial feedforward weights $\mathbf{F}$ and recurrent weights $\mathbf{\Omega}$.

We therefore need to make a small change in perspective: instead of fixing a decoder upfront we require the network to perform a double-minimization with respect to both the spike times $\mathbf{o}$ as well as with respect to the decoder $\mathbf{D}$,

$$L^* = \min_{\mathbf{o},\mathbf{D}} \left\langle \|\mathbf{x} - \mathbf{D}\mathbf{r}\|^2 + \mu \|\mathbf{r}\|^2 + \nu \|\mathbf{r}\|_1 \right\rangle \text{ s.t. } \|\mathbf{D}_n\|_2^2 = 2T_n - \mu - \nu, \quad \text{(S.16)}$$

[8] Strictly speaking, the spike trains $\mathbf{o}$ are obtained through a greedy minimization of the time-dependent loss $\ell(t)$. In practice, however, this approximation works quite well [1]

[9] Note that this decoder might even be chosen very badly and result in pathological cases such as one neuron firing with extremely high rates.

where the constraint on the decoder length arises from (S.14). This minimization defines an optimal decoder, $\mathbf{D}^*$, whose precise form will depend on the distribution of input signals $q(\mathbf{x})$.

2.3 The problem of learning the synaptic weights

The objective function (S.16) seems to pose a baffling problem to solve with a neural network for multiple reasons: first, at the beginning of learning the decoder is not explicitly represented in the network connectivity, so how could the network 'learn' a decoder? Second, neither the feedforward nor the recurrent weights are explicitly part of the objective, (S.16). Rather, the instantaneous firing rates, $\mathbf{r}(t)$, depend implicitly on the connectivity. Hence, even a simple gradient descent with respect to the synaptic weights is not possible. Third, even if the feedforward weights are initially set to the correct values, i.e., to the values of the optimal decoder $\mathbf{D}^*$, i.e. $\mathbf{F} = \mathbf{D}^{*\top}$, we could still not learn the desired recurrent connectivity $\mathbf{\Omega} = -\mathbf{F}\mathbf{F}^\top - \mu\mathbf{I}$. The key problem is that neurons have no direct access to their respective feedforward weights. More precisely, the input to the network is given by $\mathbf{F}\mathbf{x}$, but from the perspective of the network $\mathbf{F}$ and $\mathbf{x}$ are only defined up to a linear orthogonal transformation $\mathbf{A}$, $\mathbf{F}\mathbf{x} = (\mathbf{F}\mathbf{A}^\top)(\mathbf{A}\mathbf{x})$. Before showing how we can address these problems, we will briefly discuss a fourth issue, concerning the thresholds T_n of the neurons and their relation to the length of the decoder, as well as the feedforward and recurrent connectivities.

2.4 The problem of the appropriate decoder scale

Section 1.7 showed that the thresholds, T_n are intimately linked to the length of the decoder weights. To address this problem, we introduced a constraint on the length of the decoders in the double-minimization (S.16). In practice, we will therefore assume that the thresholds, T_n, are constant throughout learning and then re-interpret (S.14) as imposing a constraint on the length of the decoder weights,

$$\|\mathbf{D}_n\|_2^2 = 2T_n - \mu - \nu.$$

In turn, this 'scaling constraint' will effect the scaling of both the feedforward and recurrent weights. Using our knowledge of the optimal architecture, $\mathbf{F}^\top = \mathbf{D}$ and $\mathbf{\Omega} = -\mathbf{D}^\top\mathbf{D} - \mu\mathbf{I}$, we can directly interpret the constraint on the decoder as constraints on the feedforward and recurrent connectivity, namely

$$\begin{aligned} \|\mathbf{F}_n\|_2^2 &= 2T_n - \mu - \nu, \\ -\Omega_{nn} = \|\mathbf{D}_n\|_2^2 + \mu &= 2T_n - \nu. \end{aligned}$$

These scaling constraints need to be taken into account when learning the synaptic connectivity. However, it is important to note that the constraints only change the scale but not the structure of the network connectivity. For this reason, and in order to concentrate on the core intuitions behind the learning rule, we will delay the treatment of the scaling problem until section 4.7.

3 The core intuitions behind learning

The last section showed that we need to exploit some other property of the network in order to be able to learn the desired connectivity. In this section, we will explain the core intuitions behind our learning rules and show why the balance between excitation and inhibition is tied to the quality of the output code and to the desired network architecture. In order to simplify things, we will neglect the linear cost terms. Hence, while this section does not consider the most general scenario (with both L1 and L2 costs and correct scaling), we will revisit this issue in section 4, where the actual learning rules used in the main paper are derived.

3.1 Error-driven coding

The 'locality of information' constraint suggests that the learning problem should be attacked from the point of view of the single neuron. The membrane voltage of one of our neurons will initially obey (S.6), which we here re-express for the n-th neuron,

$$V_n = \mathbf{F}_n^\top \mathbf{x} + \mathbf{\Omega}_n^\top \mathbf{r},$$

where the first term is the feedforward input and the second term the recurrent input.

To understand what our neuron should do with its synapses, we will first consider what happens at the level of this neuron, once the input signal can be properly reconstructed from the network output with some decoder $\mathbf{D}$. We emphasize that we do not assume that we know the shape of $\mathbf{D}$ or that we are already in the optimal architecture (Section 1.7). Rather, we will simply assume that the network is in *some* state in which *some* decoder $\mathbf{D}$ will properly do the job, in which case the reconstruction error should be very small, so that $\mathbf{x} - \hat{\mathbf{x}} \approx 0$.

From the point of view of the n-th neuron, this reconstruction error is inaccessible. Indeed, our neuron only receives a small part of the input signal, namely the input signal as seen through the lense of its feedforward weights, $\mathbf{F}_n^\top \mathbf{x}$. However, the reconstruction error ε_n for this part of the input signal should, of course, likewise be close to zero so that

$$\begin{aligned} \varepsilon_n &= \mathbf{F}_n^\top \mathbf{x} - \mathbf{F}_n^\top \hat{\mathbf{x}} \\ &= \mathbf{F}_n^\top \mathbf{x} - \mathbf{F}_n^\top \mathbf{D}\mathbf{r} \\ &\approx 0. \end{aligned}$$

Our key insight is now that this latter equation will be identical to the voltage equation if we assume that $\mathbf{\Omega}_n^\top = -\mathbf{F}_n^\top \mathbf{D}$ and $\varepsilon_n = V_n \approx 0$. Hence, we have obtained two *sufficient* conditions for the network to properly represent the input signals. If we can furthermore ensure that the feedforward weights align with the decoder, and that $\mathbf{D} = \mathbf{D}^*$, then we have learnt the optimal architecture from Section 1.7, and found the minimum of the loss function (S.16).

3.2 The four conditions of learning

These insights lead us to the following *four conditions* of learning:

1. The membrane voltage of each neuron should remain close to zero, i.e., its resting potential. We can interpret this to mean that the membrane voltage fluctuations should be minimized or bounded as tightly as possible. Accordingly, any deviation from rest caused by the feedforward inputs must be immediately eliminated by the recurrent inputs. In other words, the feedforward and recurrent inputs into each cell need to balance each other on short time-scales. As a consequence, any excitatory input into the cells must be quickly canceled by an inhibitory input of equal size, a condition known as tight EI balance.

2. The recurrent connectivity should be of the form $\mathbf{\Omega} = -\mathbf{FD}$ where $\mathbf{D}$ is an a priori unknown decoder matrix. As a consequence, the membrane voltage of each neuron can be interpreted as a *projection of the reconstruction error*, ε_n, which we will refer to as 'error-driven coding.' Indeed, the recovery of the reconstruction error in the membrane voltages is a key ingredient of the optimal network, see equation (S.15). We emphasize that the decoder matrix, $\mathbf{D}$, is unspecified at this point, and that not all matrices $\mathbf{D}$ will allow a network to fulfill condition (1), as well. As a consequence, the target $\mathbf{\Omega} = -\mathbf{FD}$ consists of a large, if unspecified, set of possible matrices, and 'error-driven coding' can be achieved by a large set of networks.

3. The network architecture that we have derived so far deviates from the optimal architecture, since $\mathbf{F}$ and $\mathbf{D}^\top$ are not necessarily the same matrices. Accordingly, we need to somehow make sure that the feedforward weights $\mathbf{F}$ align with the (unknown) decoder $\mathbf{D}^\top$.

4. All of these conditions take the point of view of the single neuron. To make sure that the network as a whole represents the input signals properly, the feedforward weights need to properly span the space of input signals. If, for instance, the feedforward weights of all neurons were identical, the network could at most represent the one-dimensional space spanned by these feedforward weights—a pathological and uninteresting solution. We can make sure that the signal space is properly covered if both $\mathbf{D}$ and $\mathbf{F}$ converge to the global optimum $\mathbf{D}^*$ of (S.16).

These conditions suggest a specific program for learning the synaptic weights. First, starting from random feedforward and recurrent weights, we need to learn a balanced system in which the recurrent weights converge to a low-rank solution, $\mathbf{\Omega} \to -\mathbf{FD}$. In a second step, we can then aim to tighten the balance between excitatory and inhibitory currents by aligning $\mathbf{D}$ and $\mathbf{F}$ such that both converge to the global optimum, $\mathbf{D}^*$.[10]

In the following two subsections we derive a local learning rule for the recurrent synapses for which the fixed points obey properties (1) and (2). For an even tighter balance, we then introduce a learning mechanism for the feedforward synapses that will make the network structure converge to the solution (S.13) of the quadratic optimization problem (S.16).

[10] We re-emphasize that the decoder $\mathbf{D}$ is *not* a biophysical quantity of the network. However, it does serve as an important *conceptual* tool that is central to the development of the learning rules.

3.3 Condition 1: Recurrent weights learn to balance feedforward inputs spike by spike

The shortest possible time-scale at which a single spiking neuron can be balanced is limited by the interval between any two consecutive spikes of the *population*. We will refer to this interval as a population interspike interval (pISI), in contrast to the standard interspike interval (ISI) that is defined for two consecutive spikes of the *same* neuron.

We illustrate this idea in Fig. 1B in the main text. Here a cell receives excitatory feedforward inputs and inhibitory spikes from three pre-synaptic neurons. Between the second and the third spike (gray area) the cell integrates its feedforward input currents and depolarizes its membrane voltage. The arrival of the second inhibitory spike (red) then causes a hyperpolarization of the membrane potential (see voltage trace in middle panels). In the balanced case (left column) the hyperpolarization due to the inhibitory spike from the red neuron perfectly cancels the depolarization through the excitatory feedforward connections (gray area).

To understand how to balance a single cell on such a short time-scale, we rewrite the membrane potential as a sum over spikes. We index the spikes in the network by the time of their occurrence, writing $t_1, t_2, \ldots$ for the successive spike times of the population. We then introduce a second index in order to identify which neuron fired a particular spike, writing $k(i)$ to indicate that the i-th spike, t_i, was fired by the k-th neuron. With this notation in mind, let us define the integral over the input signal $\mathbf{c}(t)$ in the interval between two consecutive population spikes at time t_{i-1} and t_i:

$$\mathbf{g}(t_i) := \int_{t_{i-1}}^{t_i} d\tau\, \mathbf{c}(\tau) e^{-\lambda(t_i-\tau)}.$$

We can then write the membrane voltage V_n at time t_i (i.e., at the time of the i-th spike) as (confer (S.6))

$$\begin{aligned}
V_n(t_i) &= \mathbf{F}_n^\top \mathbf{x}(t_i) + \mathbf{\Omega}_n^\top \mathbf{r}(t_i) \\
&= \mathbf{F}_n^\top \int_0^\infty d\tau\, \mathbf{c}(\tau) e^{-\lambda(t_i-\tau)} + \mathbf{\Omega}_n^\top \int_0^\infty d\tau\, \mathbf{o}(\tau) e^{-\lambda(t_i-\tau)} \\
&= \sum_{j\le i} \mathbf{F}_n^\top \mathbf{g}(t_j) e^{-\lambda(t_i-t_j)} + \sum_{j\le i} \Omega_{nk(j)} e^{-\lambda(t_i-t_j)} \\
&= \sum_{i\le j} (\mathbf{F}_n^\top \mathbf{g}(t_j) + \Omega_{nk(j)}) e^{-\lambda(t_i-t_j)},
\end{aligned}$$

where $k(j)$ denotes the index of the neuron spiking at time t_j, as explained above. Here $\mathbf{F}_n \mathbf{g}(t_j)$ corresponds to the accumulated excitatory current during the pISI before the j-th spike, i.e., the total charge transfer. In turn, $\Omega_{nk(j)}$ corresponds to the immediate inhibitory charge transfer caused by the j-th spike itself. The network is perfectly balanced on the shortest time scale if these two opposing charge transfers cancel exactly. In more practical terms, the network is balanced on the shortest time scale if the (squared) net charge transfer, $\left(\mathbf{F}_n^\top \mathbf{g}(t_j) + \Omega_{nk(j)}\right)^2$, is as small as possible.

We hence concentrate on minimizing the objective

$$L = \sum_{n,i} \left(\mathbf{F}_n^\top \mathbf{g}(t_i) + \Omega_{nk(i)} \right)^2 ,$$

where the sum runs over both neurons, n, and spike times, i. We can minimize this objective by updating the recurrent synaptic weights after each spike in a greedy manner,

$$\Delta\Omega_{nk}(t) \propto \begin{cases} -\mathbf{F}_n^\top \mathbf{g}(t) - \Omega_{nk}(t) & \text{when neuron } k \text{ spikes,} \\ 0 & \text{otherwise.} \end{cases} \tag{S.21}$$

This rule has a rather intuitive meaning, as explained in the main paper and illustrated in Fig 1B: if the excitation a neuron receives in the last pISI is higher than the subsequent lateral inhibition, inhibition is strengthened, and vice versa[11]. Importantly, the learning rule relies only on local information, i.e. on quantities that are available to the neuron under investigation.

3.4 Condition 2: Spike-by-spike balance results in error-driven coding

By using the above learning rule for the recurrent weights, we can establish spike-by-spike balance—condition (1) in Section 3.1—for every single neuron $n = 1 \dots N$. We will now show that error-driven coding—condition (2)—comes out as a by-product of this learning rule.

The proof is straightforward. We simply investigate the fixed points of the learning rule (S.21), i.e. all points for which the mean update is zero, $\langle \Delta\mathbf{\Omega} \rangle = 0$. We obtain

$$\Omega_{nk} = -\left\langle \mathbf{F}_n^\top \mathbf{g} \right\rangle_{k \text{ spikes}} = -\mathbf{F}_n^\top \langle \mathbf{g} \rangle_{k \text{ spikes}} ,$$

where the brackets denote an average taken over all the spike-times of neuron k. This formula has two interesting consequences. First, the strength of the inhibitory synapse from neuron k to neuron n equals the average excitatory feedforward current that neuron n receives in the pISI before a spike of neuron k. Thus, all neurons in the population cooperate to keep the voltage of the post-synaptic cell as constant as possible. Second, the fixed points for the recurrent weights can be written as $\Omega_{nk} = -\mathbf{F}_n^\top \mathbf{D}_k$ where[12]

$$\mathbf{D}_k = \langle \mathbf{g} \rangle_{k \text{ spikes}} . \tag{S.22}$$

We note that this is the desired low-rank factorization for "error-driven coding", i.e., $\mathbf{\Omega} \to -\mathbf{FD}$. In this regime the membrane voltage of each cell tracks a projection of the error, and so the network fulfills condition (2) in section 3.1.

To summarize, we derived a simple learning rule for the recurrent connections from the principle that each recurrently fired spike should balance the

[11] Note once more that, for illustrative purposes, we here suppose that the recurrent weights are inhibitory and all feedforward weights are excitatory. We use this simplified picture for the rest of the SI. In the case described here, in which the network violates Dale's law, feedforward and recurrent weights can have both positive as well as negative signs.

[12] Please note that $\mathbf{F}_k$ corresponds to the k-th *row* of matrix $\mathbf{F}$ while $\mathbf{D}_k$ corresponds to the k-th *column* of matrix $\mathbf{D}$. Nonetheless, all vectors in the SI, including these two, are assumed to be column vectors.

feedforward input of its respective postsynaptic cells. This rule seeks to balance the network on the shortest possible time scale and thereby yields the desired low-rank factorization of the recurrent weights which is important for error-driven coding. Furthermore, we have derived an explicit formula for the decoder. Hence, even though the decoder was initially unknown, and even though the decoder does not have a direct biophysical manifestation, it can be computed through biophysical quantities, namely, the input signal sampled at the spikes of the different neurons.

3.5 Interlude: The importance of quadratic costs

The 'error-driven coding' architecture, $\mathbf{\Omega} = -\mathbf{FD}$, achieves the primary objective, i.e. representing the signal $\mathbf{x}(t)$, which can now be read out via the decoder $\mathbf{D}$. Moreover, the voltages of the neurons now represent part of the global coding error, or, more generally, the loss function.

However, even thought the loss function is now represented within the network, it is not yet minimized. More specifically, the efficiency of the representation depends strongly on the exact choice of the feedforward weights. There are two problems. First, if the feedforward weights do not cover some part of the input space (as in Figure 2, central column), then the reconstruction cost can still be high in that part of the space. Second, even if the feedforward weights cover the whole space, so that $\hat{\mathbf{x}} \approx \mathbf{x}$ everywhere, the particular spiking code chosen by the system can still be wildly inefficient: since we have not considered any cost terms, neurons could fire at very high rates in order to properly represent the signal. To find a better distribution of the feedforward weights, we therefore need to first re-introduce the L2 cost term, i.e., the cost term that severely punishes high firing of individual neurons.

In the presence of an L2 cost, we know the form of the optimal recurrent connectivity from section 1.7. Adapted to the error-driven coding architecture, the recurrent weights should therefore converge to $\mathbf{\Omega} = -\mathbf{FD} - \mu\mathbf{I}$. We can achieve this new fixed point of the recurrent learning rule (S.21) by introducing a small regularization term, $\mu\delta_{ij}$, so that

$$\Delta\Omega_{nk}(t) \propto \begin{cases} -\mathbf{F}_n^\top \mathbf{g}(t) - \Omega_{nk}(t) - \mu\delta_{nk} & \text{when neuron } k \text{ spikes,} \\ 0 & \text{otherwise.} \end{cases} \tag{S.23}$$

Importantly, the learning rule will still seek to balance the system as tightly as possible given the extra constraints. Following the logic of the previous section, one can see that $\mathbf{\Omega}$ will converge to the desired fixed point. After convergence of the recurrent weights to $\mathbf{\Omega} = -\mathbf{FD} - \mu\mathbf{I}$, the membrane potential of each neuron can be written as

$$\mathbf{V} = \mathbf{F}(\mathbf{x} - \mathbf{Dr}) - \mu\mathbf{r}.$$

As a side note, we point out that the introduction of the cost term does not alter the definition of the decoder (S.22).

3.6 Condition 3: Feedforward weights learn to mimic the decoder

The derivation of the optimal network, section 1.7, suggests that $\mathbf{F}$ should eventually align with $\mathbf{D}^\top$, as explained in condition (3) in section 3.1. Ideally, one

would therefore want an update rule of the form $\Delta\mathbf{F} \propto \mathbf{D}^\top - \mathbf{F}$, which would move the feedforward weights towards the decoder $\mathbf{D}^\top$. Unfortunately, this learning rule is not biophysically realistic since $\mathbf{D}$ is not an explicit quantity in the network. However, using the fixed-point equation for the decoder, (S.22), we can replace $\mathbf{D}$ to obtain a local, biophysical rule

$$\Delta\mathbf{F}_n(t) \propto \begin{cases} \mathbf{g}(t) - \mathbf{F}_n(t) & \text{when neuron } k \text{ spikes,} \\ 0 & \text{otherwise.} \end{cases}$$

If the learning of the feedforward connectivity occurs more slowly than the learning of the recurrent connectivity, then all fixed points of the feedforward network connectivity will fulfill $\mathbf{F}_n = \langle \mathbf{g} \rangle_{n \text{ spikes}} = \mathbf{D}_n$ and hence $\mathbf{F} \to \mathbf{D}^\top$ as desired. From a biophysical perspective the first term in the learning rule, $\langle \mathbf{g} \rangle_{n \text{ spikes}}$, corresponds to the average input signal integrated before a spike of neuron n. Such a signal could be computed in the presynaptic terminal, for instance, which, given the complex machinery of synaptic plasticity is well within the realm of possibilities.

We make two observations about the feedforward rule. First, the rule will only change the feedforward weights if a postsynaptic spike (of neuron n) coincides with a previous presynaptic input (the integrated input signal up to the time of the postsynaptic spike). In other words, this learning rule corresponds to the causal part of the standard STDP-rule (see Fig 3Ai in the main text). Second, a neuron that never spikes will not change its feedforward weight. This latter scenario is problematic since the neuron is then essentially lost to the network. However, it can be avoided either by introducing a noise term in the learning rule, or by lowering the neuron's threshold. We used this latter solution to overcome this problem in the initial stages of learning for Figure 4, see also Section 6.5.

3.7 Condition 4: Learning rules minimize loss function

While the feedforward learning rule shapes the connectivity into the desired form (section 1.7), it is not a priori clear whether these changes also help to minimize the loss function, (S.16), which was our fourth condition on learning. We will now show that the learning rule derived in the previous section achieves exactly that. To do so, we will investigate how the learning rules affect the (average) voltages, since the voltages are directly linked to the reconstruction errors and thereby the loss function. To keep things simple, we will assume that the recurrent connectivity is already learnt, and we will write $\Delta\mathbf{F} \propto \mathbf{D}^\top - \mathbf{F}$ for the feedforward update. The resulting change in the voltage will then—on average—be proportional to

$$\begin{aligned} \Delta\mathbf{V} &= \big\langle \Delta\mathbf{F}(\mathbf{x} - \mathbf{D}\mathbf{r}) \big\rangle, \\ &\propto \big\langle (\mathbf{D}^\top - \mathbf{F})(\mathbf{x} - \mathbf{D}\mathbf{r}) \big\rangle, \\ &= \big\langle \mathbf{D}^\top(\mathbf{x} - \mathbf{D}\mathbf{r}) - \mu\mathbf{r} \big\rangle - \big\langle \mathbf{F}(\mathbf{x} - \mathbf{D}\mathbf{r}) - \mu\mathbf{r} \big\rangle \\ &= \big\langle \mathbf{D}^\top(\mathbf{x} - \mathbf{D}\mathbf{r}) - \mu\mathbf{r} \big\rangle - \big\langle \mathbf{V} \big\rangle. \end{aligned}$$

Since the learning of $\mathbf{F}$ happens on a much slower time-scale than the learning of the recurrent weights, the second term on the r.h.s., i.e., the average voltage, will be close to zero, as the network remains in a tightly balanced state

throughout the learning of the feedforward weights. Accordingly, the first term will dominate changes in the voltage so that

$$\begin{aligned}\Delta \mathbf{V} &\propto \left\langle \mathbf{D}^\top(\mathbf{x} - \mathbf{D}\mathbf{r}) - \mu \mathbf{r} \right\rangle, \\ &\propto -\frac{\partial L}{\partial \mathbf{r}}\end{aligned}$$

where L is the averaged loss function (S.8) in the absence of the linear cost term. In other words, the change in voltage will push the instantaneous firing rate of the network in the direction of the antigradient of the loss function, thus minimising it.

3.8 Conclusions and biological realism reconsidered

To summarize, in this section we illustrated how an STDP-like learning rule for the feedforward connections in conjunction with a recurrent learning rule that seeks to tightly balance excitatory and inhibitory inputs, leads to a network architecture that optimizes the average loss, (S.8), and thereby produces an efficient spike code of the input signals. The derived learning rules are local, in that they only require knowledge of quantities that are available to the neurons.

There are several issues that we did not consider so far. First, we did not study linear costs or non-whitenend input distributions. Second, even though the learning rules are local, their biological plausibility may still be questioned, since the rules require the integration of input currents between successive spikes of the population. While it cannot be ruled out that actual neurons (or synapses) do keep track of these quantities, it has not been observed, either. Third, we ignored the scaling relations between the thresholds and the synaptic connectivities, derived in Section 2.4. In the next section, we address these concerns and derive learning rules based on the voltages of the neurons. We furthermore consider the extension to the full EI network.

4 The Voltage-based learning rules

In this section, we will derive the learning rules described in the main text. The derivations follow the spirit of the last section. To recapitulate, in order to improve the ability of our network to properly encode the input signals, we need to satisfy four conditions. First, the recurrent weights of each neuron need to learn to balance the feedforward inputs spike by spike. Second, the recurrent connectivity needs to converge to $\mathbf{\Omega} \to -\mathbf{F}\mathbf{D} - \mu\mathbf{I}$ for a suitable decoder $\mathbf{D}$. Third, the membrane fluctuations need to be further tightened by moving the feedforward weights to $\mathbf{F} \to \mathbf{D}^\top$, and the recurrent weights to $\mathbf{\Omega} \to -\mathbf{D}^\top\mathbf{D} - \mu\mathbf{I}$. Fourth, we need to make sure that the joint interaction of the learning rules minimizes our global loss function.

4.1 Recurrent weights: Learning in the absence of cost terms

As in section 3.4, we start by considering the learning of the recurrent synapses without costs. The target of learning is then a balanced network with recurrent weights $\mathbf{\Omega} = -\mathbf{F}\mathbf{D}$. We showed above that it is enough to seek a balanced state

in order to reach both properties. In the same spirit, we first establish a suitable and practical measure of membrane voltage fluctuations, derive learning rules for the recurrent weights that minimize those fluctuations and then show that the network will converge to a low-rank configuration $\mathbf{\Omega} \to -\mathbf{FD}$.

A particularly straightforward way of measuring voltage fluctuations is the temporal average of the squared voltage deviations from rest $V_0 = 0$, i.e.,

$$L = \left\langle \|\mathbf{V}(t)\|^2 \right\rangle_t .$$

However, evaluating the exact voltage deviations requires a precise tracking of the membrane voltage at all times, and may thus be infeasible for real neurons. Inspired by the insights from the previous section, we start with the presumption that learning occurs only during the presence of a presynaptic spike. We can then reduce the problem to minimizing the deviations of the membrane voltages around the time of a presynaptic spike. In the optimal network, the membrane voltage of a spiking neuron jumps from the threshold, T, *before* the spike to the reset, $-T$, *after* the spike. Similarly, to achieve tight balance the membrane voltage of all other neurons should ideally jump from $+V$ before a presynaptic spike to $-V$ after the spike. This motivates the following spike-based measure of the membrane voltage fluctations,

$$L = \left\langle \left\| \frac{1}{2} \left(\mathbf{V}^{\text{before}}(t_j) + \mathbf{V}^{\text{after}}(t_j) \right) \right\|^2 \right\rangle_{\text{spikes}} \tag{S.24}$$

where t_j is the time of the j-th spike in the population and $\langle \cdot \rangle_{\text{spikes}}$ denotes the expectation value over those spikes[13]. The superscripts "before" and "after" refer to the voltage values immediately before and after a spike. The recurrent weights enter (S.24) through their effect on the post-spike membrane potential,

$$\mathbf{V}^{\text{after}}(t_j) = \mathbf{V}^{\text{before}}(t_j) + \mathbf{\Omega} \mathbf{e}_{k(j)},$$

where $k(j)$ is again the index of the neuron that spikes at time t_j and $\mathbf{e}_{k(j)}$ is a unit vector with zero entries except at position $k(j)$. The introduction of $\mathbf{e}_{k(j)}$ is a bit cumbersome but useful: it allows, for example, to write the relation between the instantaneous rates of the whole population before and after a spike of neuron k at time t_j as a simple vector equation,

$$\mathbf{r}^{\text{after}}(t_j) = \mathbf{r}^{\text{before}}(t_j) + \mathbf{e}_{k(j)}.$$

The deviation at time t_j is thus

$$\begin{aligned} L_j &= \left\| \frac{1}{2} \left(\mathbf{V}^{\text{before}}(t_j) + \mathbf{V}^{\text{after}}(t_j) \right) \right\|^2 \\ &= \left\| \mathbf{V}^{\text{before}}(t_j) + \tfrac{1}{2} \mathbf{\Omega} \mathbf{e}_{k(j)} \right\|^2 . \end{aligned}$$

[13] Note that in practice, i.e., in numerical simulations, we replace this expectation value with a moving sum over a sufficiently large number of spikes. More precisely, we set $\langle x \rangle_{\text{spikes}} = \frac{1}{J} \sum_{j=1}^{J} x(t_j)$ where t_j is the spike-time of the j-th spike in the past (relative to the current time t).

To minimize the voltage deviations, we perform a greedy optimization every time a spike was fired by one of the neurons in the network,

$$\Delta\mathbf{\Omega}(t_j) \propto -\frac{\partial L_j}{\partial \mathbf{\Omega}} \propto -\left(2\mathbf{V}^{\text{before}}(t_j) + \mathbf{\Omega}\mathbf{e}_{k(j)}\right)\mathbf{e}_{k(j)}^\top. \tag{S.25}$$

More explicitely, the weight Ω_{nk} is updated at every spike of the presynaptic neuron $k = k(j)$ such that the deviation of the postsynaptic membrane voltage from rest is minimized,

$$\Delta\Omega_{nk}(t_j) \propto \begin{cases} -2V_n^{\text{before}}(t_j) - \Omega_{nk} & \text{if } k \text{ spiked,} \\ 0 & \text{otherwise.} \end{cases} \tag{S.26}$$

This learning rule differs from the one in the last section mainly in the first term on the r.h.s.: instead of compensating for the integrated feedforward current during the last pISI, the synapses here learn to compensate for the deviation of the membrane voltage from rest.

4.2 Recurrent weights: Fixed point analysis

In this section we show that the fixed-points (i.e. the points at which the mean change in the recurrent weights is zero) of the learning rule (S.26) are of the desired low-rank configuration $\mathbf{\Omega} \to -\mathbf{FD}$. In the next section we will then prove that the system will globally converge to one of these fixed-points under mild assumptions.

Mathematically, the fixed-points are defined as those recurrent weights for which $\langle\Delta\mathbf{\Omega}\rangle_{\text{spikes}} = 0$. All quantities below, such as $\mathbf{V}(t_j)$, $\mathbf{x}(t_j)$, etc., are to be understood as immediately *before* a spike, and we hence drop the superscript "before" for the rest of the SI, as well as the explicit reference to the spike time, t_j, for ease of notation[14]. The learning rule, (S.25), can then be rewritten as

$$\begin{aligned} \Delta\mathbf{\Omega} &\propto -2\mathbf{V}\mathbf{e}_{k(j)}^\top - \mathbf{\Omega}\mathbf{e}_{k(j)}\mathbf{e}_{k(j)}^\top \\ &\overset{\text{(S.6)}}{=} -2(\mathbf{F}\mathbf{x} + \mathbf{\Omega}\mathbf{r})\mathbf{e}_{k(j)}^\top - \mathbf{\Omega}\mathbf{e}_{k(j)}\mathbf{e}_{k(j)}^\top \\ &= -2\mathbf{F}\mathbf{x}\mathbf{e}_{k(j)}^\top - \mathbf{\Omega}\left(2\mathbf{r} + \mathbf{e}_{k(j)}\right)\mathbf{e}_{k(j)}^\top. \end{aligned} \tag{S.27}$$

To investigate the fixed points, we need to study the effect of applying this learning rule repeatedly, i.e., over many spike times t_j. From (S.27) and the fixed-point condition $\langle\Delta\mathbf{\Omega}\rangle_{\text{spikes}} = 0$ we obtain the defining property of the fixed points of the recurrent weights,

$$2\mathbf{F}\left\langle \mathbf{x}\mathbf{e}_{k(j)}^\top \right\rangle_{\text{spikes}} = -\mathbf{\Omega}\left\langle \left(2\mathbf{r} + \mathbf{e}_{k(j)}\right)\mathbf{e}_{k(j)}^\top \right\rangle_{\text{spikes}}.$$

Under the mild condition that the sum on the r.h.s. has full rank we can directly infer that any fixed point of $\mathbf{\Omega}$ is of the form $-\mathbf{FD}$ where $\mathbf{D}$ is (implicitly) defined by

$$2\left\langle \mathbf{x}\mathbf{e}_{k(j)}^\top \right\rangle_{\text{spikes}} = \mathbf{D}\left\langle \left(2\mathbf{r} + \mathbf{e}_{k(j)}\right)\mathbf{e}_{k(j)}^\top \right\rangle_{\text{spikes}}. \tag{S.28}$$

[14] We note that the input signal $\mathbf{x}(t_j)$ does not jump at the time t_j of the presynaptic spike, hence the distinction between before and after is irrelevant for this quantity.

While the matrix $\mathbf{D}$ can be interpreted as a linear decoder, which one could use to reconstruct the input signal from the spike trains, it is not explicitly realized within the network, since it is merged into the recurrent weights and arises dynamically through learning. In other words, the decoder is not defined upfront but the recurrent connectivity converges to a low-rank factorization from which an external observer can read off the linear decoder.

Note that (S.28) is a matrix equation with dimensions $I \times N$. To understand the exact nature of the arising decoder, $\mathbf{D}$, it is instructive to look at each element i, n individually,

$$\begin{aligned} 2\left\langle x_i \delta_{n,k(j)} \right\rangle_{\text{spikes}} &= \mathbf{D}_i^\top \left\langle \left(2\mathbf{r} + \mathbf{e}_{k(j)}\right) \delta_{n,k(j)} \right\rangle_{\text{spikes}}, \\ \Leftrightarrow 2\left\langle x_i \right\rangle_{\text{n spikes}} &= \mathbf{D}_i^\top \left\langle 2\mathbf{r} + \mathbf{e}_n \right\rangle_{\text{n spikes}}, \end{aligned}$$

where $\langle \cdot \rangle_{\text{n spikes}}$ is simply an average over all the spikes of neuron n. Using the definition for the readout, (S.7), we obtain

$$\begin{aligned} \Leftrightarrow 2\left\langle x_i \right\rangle_{\text{n spikes}} &= 2\left\langle \hat{x}_i \right\rangle_{\text{n spikes}} + D_{in}, \\ \Leftrightarrow D_{in} &= 2\left\langle x_i - \hat{x}_i \right\rangle_{\text{n spikes}}. \end{aligned} \tag{S.29}$$

Accordingly, the elements of the decoder are aligned with the reconstruction errors at the time of a spike. During learning, the optimal decoder will therefore move in directions with the largest error and hence will aim to cover as best as possible the signal space.[15]

The resulting relation for the decoder is essentially equivalent to the constraint optimal decoder derived in section 1.6, up to a scaling parameter, which we will consider further down. The minimum of the constraint loss function was found as (S.12)

$$D_{in}^* \propto \left\langle (x_i - \hat{x}_i) r_n \right\rangle_t \ .$$

Accordingly, the weighting of the error by the instantaneous firing rate mirrors the weighting of the error by the spikes in (S.29).

4.3 Recurrent weights: Convergence proof

In the last subsection we derived that all fixed points of the recurrent weights are of the form $-\mathbf{FD}$, but these fixed points might be unstable. We here prove their stability by showing that any spiking network with bounded membrane voltages will converge to the desired low-rank factorization.

To this end we split the recurrent weights $\mathbf{\Omega}$ into a part that can be described by a low-rank factorization, $\mathbf{\Omega}_F = -\mathbf{FD}$ for some $\mathbf{D}$, and a residual part, $\mathbf{\Omega}_\perp = \mathbf{\Omega} - \mathbf{\Omega}_F$. In order to show that the recurrent weights $\mathbf{\Omega}$ converge to $\mathbf{\Omega}_F$, we need to prove that the average update of $\mathbf{\Omega}$ will always decrease the norm of the residual, $\mathbf{\Omega}_\perp$.

More precisely, $\mathbf{\Omega}_F$ can be described as the projection of $\mathbf{\Omega}$ onto the image of $\mathbf{F}$, i.e. $\mathbf{\Omega}_F = \mathbf{F}\mathbf{F}^+\mathbf{\Omega}$ where the superscript "+" denotes the Moore-Penrose pseudo-inverse. Vice versa, the residual is given by $\mathbf{\Omega}_\perp = (\mathbf{I} - \mathbf{F}\mathbf{F}^+)\mathbf{\Omega}$ and so

$$\begin{aligned} \mathbf{\Omega} &= \mathbf{F}\mathbf{F}^+\mathbf{\Omega} + (\mathbf{I} - \mathbf{F}\mathbf{F}^+)\mathbf{\Omega}, \\ &\equiv \mathbf{\Omega}_F + \mathbf{\Omega}_\perp. \end{aligned}$$

[15] Note that (S.29) is a self-consistency relation since $\mathbf{D}$ is also part of $\hat{x}_i$.

The update of $\mathbf{\Omega}_\perp$ is the corresponding projection of the total update of the recurrent weights (S.27),

$$\begin{aligned}\Delta\mathbf{\Omega}_\perp &= (\mathbf{I}-\mathbf{F}\mathbf{F}^+)\Delta\mathbf{\Omega},\\ &= (\mathbf{I}-\mathbf{F}\mathbf{F}^+)\left(-2\mathbf{F}\mathbf{x}\mathbf{e}_{k(j)}^\top - \mathbf{\Omega}\left(2\mathbf{r}+\mathbf{e}_{k(j)}\right)\mathbf{e}_{k(j)}^\top\right),\\ &= -\mathbf{\Omega}_\perp\left(2\mathbf{r}+\mathbf{e}_{k(j)}\right)\mathbf{e}_{k(j)}^\top. \end{aligned} \tag{S.30}$$

where the last step follows from the relation $\mathbf{F}\mathbf{F}^+\mathbf{F}=\mathbf{F}$. In order to confirm convergence, we need to prove that the mean update $\langle\Delta\mathbf{\Omega}_\perp\rangle_{\text{spikes}}$ always decreases the norm of $\|\mathbf{\Omega}_\perp\|^2$, i.e. we need to show that

$$\begin{aligned}\|\mathbf{\Omega}_\perp\|^2 &\geq \left\|\mathbf{\Omega}_\perp+\epsilon\langle\Delta\mathbf{\Omega}_\perp\rangle_{\text{spikes}}\right\|^2,\\ &= \|\mathbf{\Omega}_\perp\|^2 + 2\epsilon\,\mathrm{tr}\left[\mathbf{\Omega}_\perp^\top\langle\Delta\mathbf{\Omega}_\perp\rangle_{\text{spikes}}\right]+\mathcal{O}(\epsilon^2),\end{aligned}$$

which results in the inequality

$$0\geq \mathrm{tr}\left[\mathbf{\Omega}_\perp^\top\langle\Delta\mathbf{\Omega}_\perp\rangle_{\text{spikes}}\right]. \tag{S.31}$$

Plugging in (S.30) we find

$$\begin{aligned}\mathrm{tr}\left[\mathbf{\Omega}_\perp^\top\langle\Delta\mathbf{\Omega}_\perp\rangle_{\text{spikes}}\right] &= -\mathrm{tr}\left[\mathbf{\Omega}_\perp^\top\left\langle\mathbf{\Omega}_\perp\left(2\mathbf{r}+\mathbf{e}_{k(j)}\right)\mathbf{e}_{k(j)}^\top\right\rangle_{\text{spikes}}\right],\\ &= -\mathrm{tr}\left[\mathbf{\Omega}_\perp\left\langle\left(2\mathbf{r}+\mathbf{e}_{k(j)}\right)\mathbf{e}_{k(j)}^\top\right\rangle_{\text{spikes}}\mathbf{\Omega}_\perp^\top\right],\\ &\approx -\mathrm{tr}\left[\mathbf{\Omega}_\perp\left\langle 2\mathbf{r}\mathbf{r}^\top/|\mathbf{r}|+\mathbf{e}_{k(j)}\mathbf{e}_{k(j)}^\top\right\rangle_{\text{spikes}}\mathbf{\Omega}_\perp^\top\right],\end{aligned} \tag{S.32}$$

where we used that for a given stimulus the rates are (to first order) fairly constant[16] over time and the average of the spike counts will be equivalent to the rates, so $\left\langle\mathbf{r}\mathbf{e}_{k(j)}^\top\right\rangle_{\text{spikes}} \approx \left\langle\mathbf{r}\mathbf{r}^\top/|\mathbf{r}|\right\rangle_{\text{spikes}}$. Finally, observe that the inner bracket of (S.32) is semi-positive definite and so we proved the desired relation $\mathrm{tr}\left[\mathbf{\Omega}_\perp^\top\langle\Delta\mathbf{\Omega}_\perp\rangle_{\text{spikes}}\right]\leq 0$. Consequently, any stable network will converge to a low-rank factorization under mild assumptions.

4.4 Recurrent weights: Learning with L2 costs

As explained in section 3.5, we need to introduce quadratic costs before considering the learning of the feedforward weights. The quadratic (L2) costs change the target connectivity to $\mathbf{\Omega}\to-\mathbf{F}\mathbf{D}-\mu\mathbf{I}$. This target can be obtained through the following learning rule, modified from (S.25),

$$\Delta\mathbf{\Omega}\propto -2\big(\mathbf{V}+\mu\mathbf{r}\big)\mathbf{e}_{k(j)}^\top-(\mathbf{\Omega}+\mu\mathbf{I})\mathbf{e}_{k(j)}\mathbf{e}_{k(j)}^\top,$$

[16] Strictly speaking, this assumption is only valid in the limit of high instantaneous firing rates. In the regime of low firing rates, however, the (positive) diagonals in the inner bracket of (S.32) dominate and so the expectation value is still likely to be semi-positive definite as required to prove relation (S.31)

or, without the burden of seeing through the matrix-vector notation,

$$\Delta\Omega_{nk} \propto \begin{cases} -2\left(V_n + \mu r_n\right) - \Omega_{nk} - \mu\delta_{nk} & \text{if } k \text{ spiked,} \\ 0 & \text{otherwise.} \end{cases}$$

We remind the reader that quantities such as the voltage or the instantaneous rate are here assumed to be evaluated directly *before* a spike of neuron k, i.e., $V_n = V_n^{\text{before}}(t_k)$ and $r_n = r_n^{\text{before}}(t_k)$. To show the fixed points of this modified learning rule, we follow the exact same analysis as in section 4.2, see (S.27),

$$\begin{aligned} \Delta\boldsymbol{\Omega} &\propto -2(\mathbf{V} + \mu\mathbf{r})\mathbf{e}_{k(j)}^\top - (\boldsymbol{\Omega} + \mu\mathbf{I})\mathbf{e}_{k(j)}\mathbf{e}_{k(j)}^\top \\ &\overset{\text{(S.6)}}{=} -2(\mathbf{F}\mathbf{x} + (\boldsymbol{\Omega} + \mu\mathbf{I})\mathbf{r})\mathbf{e}_{k(j)}^\top - (\boldsymbol{\Omega} + \mu\mathbf{I})\mathbf{e}_{k(j)}\mathbf{e}_{k(j)}^\top \\ &= -2\mathbf{F}\mathbf{x}\mathbf{e}_{k(j)}^\top - (\boldsymbol{\Omega} + \mu\mathbf{I})\left(2\mathbf{r} + \mathbf{e}_{k(j)}\right)\mathbf{e}_{k(j)}^\top. \end{aligned}$$

Compared to (S.27) we only replaced $\boldsymbol{\Omega}$ by $\boldsymbol{\Omega}+\mu\mathbf{I}$. Consequently, all arguments concerning the fixed points $-\mathbf{FD}$ and convergence in section 4.2 and 4.3 now hold for $\boldsymbol{\Omega}+\mu\mathbf{I}$, and so we proved $\boldsymbol{\Omega} \to -\mathbf{FD}-\mu\mathbf{I}$. Following the same argument, the fixed points (S.29) of the decoder do not change.

4.5 Feedforward weights: Learning rule

In order to solve the full quadratic optimization problem (S.8) we need to ensure that the feedforward weights $\mathbf{F}$ align with the decoder $\mathbf{D}^\top$. To this end, we remind the reader that the decoder will converge to (S.29),

$$\mathbf{D}_n = 2\left\langle \mathbf{x} - \hat{\mathbf{x}} \right\rangle_{\text{n spikes}}.$$

In principle we would like to use this quantity to guide learning of the feedforward weights $\mathbf{F}$, just as we did in section 3.6. From a biophysical point of view, however, we cannot assume that the feedforward weights have access to the error $\mathbf{x} - \hat{\mathbf{x}}$ (only to projections of the error). Fortunately the difference between $\mathbf{x} - \hat{\mathbf{x}}$ will be proportional to the input signal $\mathbf{x}$, on average, since we assumed that the quadratic costs are non-negligible (see previous section). These costs will prohibit $\hat{\mathbf{x}}$ to fully match the size of $\mathbf{x}$, an effect that increases linearly with the size of $\mathbf{x}$. Accordingly, input signal and error are, on average, proportional to each other, i.e., $\mathbf{x} - \hat{\mathbf{x}} \propto \mathbf{x}$. The learning rule for the feedforward connections can therefore be approximated by:

$$\Delta\mathbf{F}_n(t_j) \propto \begin{cases} \mathbf{x}(t_j) - \mathbf{F}_n & \text{if } n \text{ spiked,} \\ 0 & \text{otherwise.} \end{cases} \tag{S.33}$$

Importantly, we note that, if the current $\mathbf{c}$ is changing slowly compared to the dynamics of the network, any sufficiently leaky integration of $\mathbf{c}$ is a good approximation of $\mathbf{x} - \hat{\mathbf{x}}$. This observation becomes particularly important in the case of faster inputs: here the error $\mathbf{x} - \hat{\mathbf{x}}$ can become dominated by the inability of the network to follow the inputs, so that a stable equilibrium is never reached. In such cases we often find numerically that a less leaky integration of the current leads to more efficient networks and better reconstruction errors. For the sake of mathematical precision, we will here make the assumption that $\mathbf{c}$ is changing slowly, as also stated at the very beginning, section 1.1, and we will proceed with (S.33).

4.6 Feedforward weights: Fixed-point analysis

Analysing the stable fixed points from the interacting feedforward and recurrent synaptic plasticity rules is daunting since the membrane voltage of each cell depends on the exact sequence and timing of the spikes. Under these conditions there is little we can do beyond the numerical simulations (see main text). For large networks, however, even small noise sources will considerably randomise the timings and sequences of spikes [1], and so in this limit it makes sense to analyse the fixed points under the assumption that spikes are distributed according to an inhomogeneous Poisson process with mean firing rates $\bar{r}_k(\mathbf{x})$. In this case the fixed point of the feedforward learning rule, (S.33), is simply given by computing the expectation value over stimuli,

$$\mathbf{F}^* = \left\langle \bar{\mathbf{r}} \mathbf{x}^\top \right\rangle .$$

Since the feedforward weights align with the decoder by design, the latter has the same fixed point (up to a transpose), so that

$$\mathbf{D}^* = \left\langle \mathbf{x} \bar{\mathbf{r}}^\top \right\rangle .$$

By multiplying with $\mathbf{D}^{*\top}\mathbf{D}^*$ from the right we can identify a simple condition on the fixed point,

$$\begin{aligned} \mathbf{D}^*\mathbf{D}^{*\top}\mathbf{D}^* &= \left\langle \mathbf{x} \bar{\mathbf{r}}^\top \right\rangle \mathbf{D}^{*\top}\mathbf{D}^* \\ &= \left\langle \mathbf{x} (\mathbf{D}^* \bar{\mathbf{r}})^\top \right\rangle \mathbf{D}^* \\ &= \left\langle \mathbf{x}\hat{\mathbf{x}} \right\rangle \mathbf{D}^*. \end{aligned}$$

As observed above, the reconstruction $\hat{\mathbf{x}}$ will closely follow the input signal $\mathbf{x}$, only slightly scaled down due to the quadratic costs. Since the input signal was assumed to be white, $\langle \mathbf{x}\mathbf{x}^\top \rangle = \mathbf{I}$, we can conclude that $\langle \mathbf{x}\hat{\mathbf{x}}^\top \rangle \propto \mathbf{I}$. Hence,

$$\mathbf{D}^*\mathbf{D}^{*\top}\mathbf{D}^* \propto \mathbf{D}^*.$$

This condition is only fulfilled if the transpose of $\mathbf{D}^*$ is its own pseudo-inverse, and so $\mathbf{D}^*$ is a unitary matrix. (Or, more precisely, a slightly scaled down version of a unitary matrix.) In other words, in its fixed points the network represents the independent axes of a white stimulus on orthogonal directions of the population response $\mathbf{r}$, which is optimal. We discuss the non-whitened inputs in the section 4.8.

4.7 L1 cost and the scaling problem

We have shown that under mild assumptions the combination of both learning rules will make the network converge to a $\mathbf{F} \to \mathbf{D}^\top$ and $\mathbf{\Omega} \to -\mathbf{D}^\top\mathbf{D} - \mu\mathbf{I}$, similar to the optimal connectivity structure that we derived in section 1. So far, however, we have ignored the scaling of the synaptic connectivity, i.e. the relationship between the threshold T_n and the scale of the feedforward weights $\mathbf{F}_n$ and the autapse $\mathbf{\Omega}_{nn}$, see section 2.4,

$$-\Omega_{nn} = \|\mathbf{D}_n\|_2^2 + \mu = 2T_n - \nu, \tag{S.34}$$

$$\|\mathbf{F}_n\|_2^2 = 2T_n - \mu - \nu. \tag{S.35}$$

Whereas the L2 cost modifies the recurrent connectivity, so that $\mathbf{\Omega} \to -\mathbf{D}^\top\mathbf{D} - \mu\mathbf{I}$, the L1 cost *only* enters the learning through these two equations. Since we assume that the thresholds of the neurons are given some initial value and are then never changed, our learning rules need to be modified in order to account for the appropriate scale of the synaptic weights. To guarantee the relation $-\Omega_{nn} = 2T_n - \nu$, we introduce a scaling factor β_n for every neuron n in the learning rule of the recurrent synapses,

$$\Delta\Omega_{nk} \propto \begin{cases} -\beta_n(V_n + \mu r_n) - \Omega_{nk} - \mu\delta_{nk} & \text{if } k \text{ spiked,} \\ 0 & \text{otherwise.} \end{cases} \tag{S.36}$$

Following again section 4.2 and the appropriate correction for the L2 costs in section 4.4, it is straight-forward to see that $\mathbf{\Omega}$ will converge to $-\mathbf{F}\mathbf{D} - \mu\mathbf{I}$, where the decoder $\mathbf{D}$ is now modified by the scaling factor β_n so that

$$D_{in} \to \beta_n \langle x_i - \hat{x}_i \rangle_{\text{n spikes}} .$$

Hence, in order to obey (S.34), the scaling factors β_n should evolve according to,

$$\begin{aligned} -\Omega_{nn} &= 2T_n - \nu \\ \Leftrightarrow \mathbf{F}_n^\top\mathbf{D}_n + \mu &= 2T_n - \nu, \\ \Leftrightarrow \mathbf{F}_n^\top\beta_n\langle\mathbf{x} - \hat{\mathbf{x}}\rangle_{\text{n spikes}} + \mu &= 2T_n - \nu, \\ \Leftrightarrow \beta_n &= \frac{2T_n - \mu - \nu}{\mathbf{F}_n^\top\langle\mathbf{x} - \hat{\mathbf{x}}\rangle_{\text{n spikes}}}, \\ \Leftrightarrow \beta_n &= \frac{2T_n - \mu - \nu}{T_n + \mu\langle r_n\rangle_{\text{n spikes}}} \end{aligned} \tag{S.37}$$

where in the last step we have used the relation $\mathbf{F}_n^\top\langle\mathbf{x} - \hat{\mathbf{x}}\rangle_{\text{n spikes}} = \langle V_n\rangle_{\text{n spikes}} + \mu\langle r_n\rangle_{\text{n spikes}}$ and $V_n = V_n^{\text{before}} = T_n$, since the voltage directly before the spike of the firing neuron is, by definition, the neuron's threshold. Note that in the absence of costs, $\mu = \nu = 0$, we recover $\beta_n = 2$, i.e., the unscaled learning rule (S.23).

Similarly, to guarantee the scaling $\|\mathbf{F}_n\|_2^2 = 2T_n - \mu - \nu$ of the feedforward weights, we introduce appropriate scaling factors α_n into the learning rule (S.33),

$$\Delta\mathbf{F}_n \propto \begin{cases} \alpha_n\mathbf{x} - \mathbf{F}_n & \text{if } n \text{ spiked,} \\ 0 & \text{otherwise.} \end{cases} \tag{S.38}$$

which will consequently lead to a scaling of the fixed point,

$$\mathbf{F}_n \to \alpha_n \langle\mathbf{x}\rangle_{\text{n spikes}} .$$

Hence, in order to fulfill (S.35) it should hold that

$$\begin{aligned} \|\mathbf{F}_n\|^2 &= \mathbf{F}_n^\top\mathbf{F}_n \\ &= \mathbf{F}_n^\top\alpha_n \langle\mathbf{x}\rangle_{\text{n spikes}} \\ &= 2T_n - \mu - \nu \end{aligned}$$

from which we read off an expression for the scaling factors,

$$\alpha_n = \frac{2T_n - \mu - \nu}{\mathbf{F}_n^\top \langle \mathbf{x} \rangle_{\text{n spikes}}}. \tag{S.39}$$

The learning rules (S.36) and (S.38) in conjunction with the definition of the scaling factors (S.37) and (S.39) are thus the set of rules that take into account all costs and will make the network converge to the optimal network configuration with the optimal decoding weights.

4.8 Non-whitened inputs

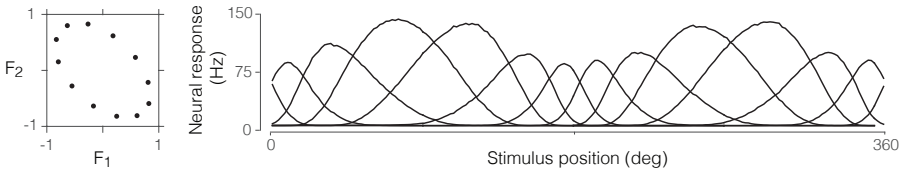

Figure S1: Example of a 12-neuron network that learns to encode two correlated input signals (with a distribution similar to Fig. 1D). (Left) The two-dimensional feedforward weights of the 12 neurons after learning. (Right) Tuning curves of all neurons in the network after training, i.e., their firing rates as a function of the angle of a two-dimensional input with constant radius in polar coordinates.

So far we have assumed that the input stimulus is zero-mean and whitened. To cover more general scenarios, we first revisit the optimal spiking neural network from section 1.7, following the approach outlined in [6] for rate networks. First, we note that a self-organized network is incapable of determining the true covariance of the signal (which could always be "arbitrarily" distorted by the feedforward weights) while the mean of the signal should be filtered out to increase efficiency (otherwise spikes are constantly emitted just to support a fixed offset). To take both aspects into account, we modify the loss function (S.10),

$$\ell = (\mathbf{x}_c - \mathbf{D}\mathbf{r})^\top \mathbf{C}^{-1} (\mathbf{x}_c - \mathbf{D}\mathbf{r}) + \mu \|\mathbf{r}\|^2 + \nu \|\mathbf{r}\|_1$$

where $\mathbf{x}_c = \mathbf{x} - \bar{\mathbf{x}}$ is the mean signal and $\mathbf{C} = \langle \mathbf{x}_c \mathbf{x}_c^\top \rangle$ is the signal covariance. In complete analogy to section 1.7, one can derive the voltages and thresholds of simple integrate-and-fire neurons,

$$\begin{aligned} \mathbf{V} &= \mathbf{D}^\top \mathbf{C}^{-1} \mathbf{x}_c - \mathbf{D}^\top \mathbf{C}^{-1} \mathbf{D}\mathbf{r} - \mu \mathbf{r}, \\ T_n &= \frac{1}{2} \left(\|\mathbf{D}_n\|^2 + \mu + \nu \right). \end{aligned}$$

Accordingly, the network is now characterized by feedforward weights $\mathbf{F} = \mathbf{D}^\top \mathbf{C}^{-1}$ and recurrent weights $\mathbf{\Omega} = -\mathbf{D}^\top \mathbf{C}^{-1} \mathbf{D} - \mu \mathbf{I} = -\mathbf{F}\mathbf{D} - \mu \mathbf{I}$. These equations show that we only need to revisit the feedforward weights, whose relation to the decoder has changed, but not the recurrent weights, whose relation to the feedforward and decoder weights remains the same. Indeed, we did not make any (implicit or explicit) assumptions on the statistics of the input in the derivation of the recurrent learning rules, and so the same learning rule (S.36) applies in this case.

To make the feedforward weights converge to $\mathbf{F} = \mathbf{D}^\top \mathbf{C}^{-1}$, we ignore the correct scaling for now (see next section) and modify the learning rule (S.33) as

$$\Delta \mathbf{F}_n \propto \begin{cases} \mathbf{x}_c - \mathbf{F}_n^\top \mathbf{x}_c \mathbf{x}_c & \text{when neuron } n \text{ spikes,} \\ -\mathbf{F}_n^\top \mathbf{x}_c \mathbf{x}_c & \text{otherwise.} \end{cases} \tag{S.40}$$

We emphasize that this learning rule is still local. We can highlight this feature by stating the learning rule for the i-th element of $\mathbf{F}_n$,

$$\Delta F_{in} \propto \begin{cases} [x_c]_i - (\mathbf{F}_n^\top \mathbf{x}_c)[x_c]_i & \text{when neuron } n \text{ spikes,} \\ -(\mathbf{F}_n^\top \mathbf{x}_c)[x_c]_i & \text{otherwise.} \end{cases}$$

Here, $\mathbf{F}_n^\top \mathbf{x}_c$ is simply the total feedforward current that the postsynaptic neuron received. Accordingly, the modified learning rule requires a multiplicative, yet local interaction between the presynaptic signal, $[x_c]_i$, and the postsynaptic current. In Fig. S1 we simulate this modified learning rule in a network of 12 neurons that receive a correlated input signal. Interestingly, the network learns tuning curves that are narrower and denser around the most frequently presented signal directions. This is reminiscent of the tuning curves derived from efficient coding principles in a population of Poisson-firing neurons [20].

Following the derivation of section 4.6, and assuming once more Poisson-distributed spike trains, the fixed points, $\mathbf{F}^*$, of the feedforward rule become

$$\begin{aligned} \left\langle \bar{\mathbf{r}} \mathbf{x}_c^\top - \mathbf{F}^* \mathbf{x}_c \mathbf{x}_c^\top \right\rangle &= 0, \\ \Leftrightarrow \mathbf{F}^* \left\langle \mathbf{x}_c \mathbf{x}_c^\top \right\rangle &= \left\langle \bar{\mathbf{r}} \mathbf{x}_c^\top \right\rangle, \\ \Rightarrow \mathbf{F}^* &= {\mathbf{D}^*}^\top \mathbf{C}^{-1}. \end{aligned}$$

where the fixed point of the decoder, $\mathbf{D}^*$ remains untouched, and becomes

$$\mathbf{D}^* \propto \left\langle \mathbf{x}_c \bar{\mathbf{r}}^\top \right\rangle.$$

Using this relation once more, and multiplying it with ${\mathbf{D}^*}^\top \mathbf{C}^{-1} \mathbf{D}^*$ from the right, we find the following relation for the decoder

$$\begin{aligned} \mathbf{D}^* {\mathbf{D}^*}^\top \mathbf{C}^{-1} \mathbf{D}^* &\propto \left\langle \mathbf{x}_c \bar{\mathbf{r}}^\top \right\rangle {\mathbf{D}^*}^\top \mathbf{C}^{-1} \mathbf{D}^*, \\ &= \left\langle \mathbf{x}_c (\mathbf{D}^* \bar{\mathbf{r}})^\top \right\rangle \mathbf{C}^{-1} \mathbf{D}^*, \\ &= \left\langle \mathbf{x}_c \hat{\mathbf{x}}_c \right\rangle \mathbf{C}^{-1} \mathbf{D}^*, \\ &\propto \mathbf{D}^*, \end{aligned}$$

which is fulfilled if $\mathbf{D}^* = \mathbf{C}^{1/2} \mathbf{U}$, where $\mathbf{U}$ is a unitary matrix. Then $\mathbf{F}^* = {\mathbf{D}^*}^\top \mathbf{C}^{-1} = \mathbf{U}^\top \mathbf{C}^{-1/2}$. This last equation exposes the solution that the network finds: it whitens the input through its feedforward filters before encoding it along orthogonal axis in the population response.

4.9 Simplifying assumptions for the main paper

The scaling factors α_n and β_n for the feedforward and recurrent weights guarantee the convergence of the network to the properly scaled weights. It is important to remember that the scaling factors only set the right scale of the

weights, they do not affect the overall structure of the optimal connectivities, $\mathbf{F} \propto \mathbf{D}^\top$ and $\mathbf{\Omega} \propto -\mathbf{D}^\top\mathbf{D} - \mu\mathbf{I}$. In practice, we set all thresholds to the same values $T_n = T$. In addition, we note that fixing the scaling factors α_n and β_n merely fixes a set of fixed points with a particular L1 cost ν and scaling of $\mathbf{D}$. To ease simulation we fix two scaling factors $\alpha = \alpha_n$ and $\beta = \beta_n$ by hand such that the fixed points exhibit reasonable cost values and scales. This approximation worked quite well. The recurrent learning rule, (S.36), then becomes

$$\Delta\Omega_{nk} \propto \begin{cases} -\beta(V_n + \mu r_n) - \Omega_{nk} - \mu\delta_{nk} & \text{if } k \text{ spiked,} \\ 0 & \text{otherwise.} \end{cases} \tag{S.41}$$

which is the equation shown in the main paper. The feedforward rule, (S.38), becomes

$$\Delta F_{in} \propto \begin{cases} \alpha x_i - F_{in} & \text{if } n \text{ spiked,} \\ 0 & \text{otherwise.} \end{cases} \tag{S.42}$$

In both rules, α and β are simply treated as free parameters.

5 Learning in the EI network

So far we have neglected Dale's law, i.e., the distinction between excitatory and inhibitory neurons. We are now going to use the intuition developed in the last three sections to explain how learning in an EI network should proceed.

Consider a population of excitatory neurons that receives feedforward inputs. Its recurrent connections are constrained to be excitatory, and so neurons with overlapping inputs cannot inhibit and thereby balance each other. If, however, a population of inhibitory neurons has learnt to represent the signal encoded by the excitatory population, then its output could in turn be used to provide the necessary balance.

We can formalize this intuition as follows. First, we consider the optimal networks from the previous sections, but without costs, so that $\mathbf{\Omega} = -\mathbf{D}\mathbf{D}^\top$. We then split the decoder weights into one part with all positive entries, $\mathbf{D}_+$, and one part with the absolute value of all negative entries, $\mathbf{D}_-$, so that $\mathbf{D} = \mathbf{D}_+ - \mathbf{D}_-$. The recurrent input into the network can then be rewritten as

$$\begin{aligned} \mathbf{\Omega r} &= -(\mathbf{D}_+ - \mathbf{D}_-)^\top(\mathbf{D}_+ - \mathbf{D}_-)\mathbf{r}, \\ &= (\mathbf{D}_-^\top\mathbf{D}_+ + \mathbf{D}_+^\top\mathbf{D}_-)\mathbf{r} - (\mathbf{D}_+^\top\mathbf{D}_+ + \mathbf{D}_-^\top\mathbf{D}_-)\mathbf{r}. \end{aligned}$$

We can identify the first term as the recurrent excitation and the second term as the recurrent inhibition. If we assume that all the neurons we have considered so far were, in fact, excitatory, and if we want these neurons to obey Dale's law, then we need to approximate the second term by means of a separate, inhibitory population. To this end, let us assume that this population of inhibitory neurons 'represents' the activity of the excitatory neurons, or, more formally, that we can retrieve an estimate of the activity of the excitatory neurons, $\hat{\mathbf{r}}$, from the activity of the inhibitory neurons, $\mathbf{s}$, via a suitable linear readout, $\tilde{\mathbf{D}}$, so that $\hat{\mathbf{r}} = \tilde{\mathbf{D}}\mathbf{s}$, In this case, we can rewrite the recurrent input into the excitatory neurons as

$$\mathbf{\Omega r} \approx (\mathbf{D}_-^\top\mathbf{D}_+ + \mathbf{D}_+^\top\mathbf{D}_-)\mathbf{r} - (\mathbf{D}_+^\top\mathbf{D}_+ + \mathbf{D}_-^\top\mathbf{D}_-)\tilde{\mathbf{D}}\mathbf{s},$$

so that the second term now derives from the inhibitory population, as desired.

To achieve this condition, we will assume that the inhibitory population minimizes the objective

$$L_I = \left\langle \left\| \mathbf{r} - \tilde{\mathbf{D}}\mathbf{s} \right\|^2 \right\rangle,$$

which is identical to the problem we have tackled in all previous sections, if we identify $\mathbf{r}$ with $\mathbf{x}$ and $\mathbf{s}$ with $\mathbf{r}$. At first, it may seem that we have simply shifted the problem of Dale's law onto the inhibitory sub-population, so that we are back where we started. However, the input into the inhibitory sub-network is now purely positive, since $\mathbf{r} \geq 0$ at all times. In turn, the (optimal) decoder $\tilde{\mathbf{D}}$ has all positive entries as well. Since the optimal subnetwork's recurrent weights are given by $\mathbf{\Omega}^{II} = -\tilde{\mathbf{D}}\tilde{\mathbf{D}}^\top$, all its recurrent weights are negative, and the inhibitory subpopulation therefore obeys Dale's law, as well.

In summary, the structure of the optimal EI network is given by the following set of feedforward and recurrent weights for the excitatory population, compare with (S.3),

$$\begin{aligned} \mathbf{F}^E &= \mathbf{D}^\top \\ \mathbf{\Omega}_{EE} &= \mathbf{D}_-^\top \mathbf{D}_+ + \mathbf{D}_+^\top \mathbf{D}_- \\ \mathbf{\Omega}_{EI} &= -(\mathbf{D}_+^\top \mathbf{D}_+ + \mathbf{D}_-^\top \mathbf{D}_-)\tilde{\mathbf{D}} \end{aligned}$$

and the following set of feedforward and recurrent weights for the inhibitory population, compare (S.4),

$$\begin{aligned} \mathbf{F}^I = \mathbf{\Omega}_{IE} &= \tilde{\mathbf{D}}^\top \\ \mathbf{\Omega}_{II} &= -\tilde{\mathbf{D}}\tilde{\mathbf{D}}^\top. \end{aligned}$$

From the derivation, we see that the excitatory population response serves as a feedforward input to the inhibitory population. Accordingly, the weights $\mathbf{\Omega}_{IE}$ are effectively feedforward and need to be trained by the standard feedforward rule, just as the feedforward weights of the excitatory population, $\mathbf{F}^E$. All other weights, i.e. $\mathbf{\Omega}_{EE}, \mathbf{\Omega}_{EI}$ and $\mathbf{\Omega}_{II}$, serve to balance either the excitatory or inhibitory populations. Accordingly, these weights must be trained using the recurrent learning rule. The training then proceeds as in the non-Dale's case except for the sign constraints on the synaptic weights.

6 Numerical Simulations

6.1 Network Dynamics

The membrane voltage V_n of each cell is simulated according to a discrete-time (Euler) approximation of the differential equations, either (S.3) and (S.4) for the EI networks, or (S.5) for the non-Dalian networks,

$$V_n(t + \Delta t) = V_n(t) + \Delta t \dot{V}_n(t).$$

In Figure 2 and 3, the target signal $\mathbf{x}(t)$ is set as follows: we first draw a random vector $\mathbf{y}(t) \in \mathbf{R}^I$ from a zero-mean Gaussian distribution $\mathbf{y}(t) \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$ for every time-step t and then convolve $\mathbf{y}(t)$ with a Gaussian kernel of size η over

time to get $\mathbf{x}(t)$. For Figure 4, the target signal was the speech spectrogram, sampled at 100Hz and interpolated to reach a temporal resolution of 0.05 ms. The input current $\mathbf{c}(t)$ is subsequently computed following (S.1), i.e.,

$$c_i(t) = \dot{x}_i(t) + \lambda x_i(t).$$

To randomize the spikes, gaussian noise terms ξ_V and ξ_T with very small variances and zero means are added respectively to the voltage equation and to the thresholds of the neurons.

In all simulations, the recurrent (and I to E) weights are trained by means of the recurrent learning rule (S.41). The feedforward weights $\mathbf{F}$ and the $E-I$ connections follow (S.42) (whitened inputs, Figure 2 and 3) or (S.40) (non-whitened input, Figure 4). However, in Figure 3 and 4, we simulated the feedforward learning rules (FF rules) using a slightly smoother version of the input signal $\mathbf{x}$. More specifically, we replaced $\mathbf{x}$ in equations (S.42) and (S.40) by the input currents integrated with a larger leak term $\lambda_F > \lambda$. The learning rule of the feedforward connection (for whitened signals) then becomes

$$\Delta\mathbf{F}_n(t_j) \propto \begin{cases} \bar{\mathbf{c}}(t_j) - \mathbf{F}_n & \text{if } n \text{ spiked,} \\ 0 & \text{otherwise.} \end{cases}$$

where $\bar{\mathbf{c}}$ obeys $\dot{\bar{\mathbf{c}}} = -\lambda_F \bar{\mathbf{c}} + \mathbf{c}$.

The constant scaling term α is chosen so as to achieve mean firing rates of around 5 to 10 Hz after training. The learning rates ϵ_F and ϵ_Ω of the feedforward and recurrent weights are either kept constant throughout the simulation (Figures 2 and 3), or progressively decreased (Figure 4). Importantly, there is a separation of time-scales such that $\epsilon_\Omega = 10\epsilon_F$; this ensures that the network is always kept in a balanced (and thus stable) regime throughout learning.

The full pseudo-code for the non-Dales case can be found in algorithm 1.

6.2 Initialization

To initialize the feedforward weights $\mathbf{F} \in \mathbb{R}^{N\times I}$, we first draw all elements from a zero-mean normal distribution, $F_{ni} \sim \mathcal{N}(0,1)$, and then normalize each row to be of length γ, i.e.

$$F_{ni} \to \gamma \frac{F_{ni}}{\sqrt{\sum_i F_{ni}^2}}.$$

In Figure 2 and 4, the initial recurrent weights $\mathbf{\Omega} \in \mathbb{R}^{N\times N}$ are proportional to the unit matrix $\mathbf{I}_N$ with proportionality ω, i.e. $\mathbf{\Omega}_0 = \omega\mathbf{I}_N$. This simplified initialization is chosen for illustrative purposes; the learning also works for more general, random initializations of the recurrent connections. The recurrent connectivity in the EI network (Figure 3) is similarly initialized as $\mathbf{\Omega}_{EE} = \omega_{EE}\mathbf{I}_{N_E}$ and $\mathbf{\Omega}_{II} = \omega_{II}\mathbf{I}_{N_I}$. In all simulations there are four times more excitatory than inhibitory neurons, $N_E = 4\cdot N_I$, and so we initialize the E-I and I-E connections according to $\mathbf{\Omega}_{EI} = \omega_{EI}[\mathbf{I}_{N_I}, \mathbf{I}_{N_I}, \mathbf{I}_{N_I}, \mathbf{I}_{N_I}]$ and $\mathbf{\Omega}_{IE} = \omega_{IE}[\mathbf{I}_{N_I}, \mathbf{I}_{N_I}, \mathbf{I}_{N_I}, \mathbf{I}_{N_I}]^\top$ where the squared brackets denote a stacking of the elements along the rows. The thresholds $T_n = T$ are kept constant (except for Figure 4, see below) over the course of the simulation (including learning) and are homogeneous across cells.

Algorithm 1 Pseudo-code for simulation of non-Dales network (Fig. 2 & 4)

procedure SIMULATION
  $N, I \leftarrow$ number of cells and input dimensions
  $\lambda \leftarrow$ membrane leak
  $\mathbf{F}(0) \leftarrow$ initial feedforward weights
  $\mathbf{\Omega}(0) \leftarrow$ initial recurrent weights
  $S \leftarrow$ total simulation time
  dt $\leftarrow$ time-step
  $\epsilon_F, \epsilon_\Omega \leftarrow$ learning rates
  $\alpha, \beta \leftarrow$ scaling factors in learning equations
  $\mu \leftarrow$ L2 cost
  $T \leftarrow$ threshold
  $\sigma, \eta \leftarrow$ standard deviation of signal, time-scale of smoothing kernel
top:
  $\mathbf{V}(0) \leftarrow \mathbf{0}$ (initial voltage)
  $\mathbf{o}(0) \leftarrow \mathbf{0}$ (initial spikes)
  $\mathbf{r}(0) \leftarrow \mathbf{0}$ (initial filtered spikes)
  $\Gamma \leftarrow$ closest integer to S/dt (number of simulation steps)
  $\mathbf{x}(\tau) \leftarrow$ drawn from $\mathcal{N}(0, \sigma^2 \mathbf{I}_I)$ for all $\tau = 1 \dots \Gamma$
  $\mathbf{x}(\tau) \leftarrow \mathbf{x}(\tau)$ filtered with Gaussian kernel of width η over time
  $\xi_V(\tau) \leftarrow$ drawn from $\mathcal{N}(0, \sigma^2_{\xi_V} \mathbf{I}_N)$ for all $\tau = 1 \dots \Gamma$
  $\xi_T(\tau) \leftarrow$ drawn from $\mathcal{N}(0, \sigma^2_{\xi_T} \mathbf{I}_N)$ for all $\tau = 1 \dots \Gamma$
loop:
  for $\tau = 1$ **to** Γ **do**
    $\mathbf{c}(\tau - 1) = \mathbf{x}(\tau) - \mathbf{x}(\tau - 1) + \lambda \,\mathrm{dt}\, \mathbf{x}(\tau - 1)$
    $\mathbf{V}(\tau) = (1 - \lambda \,\mathrm{dt})\mathbf{V}(\tau - 1) + \mathrm{dt}\, \mathbf{F}(\tau - 1)^\top \mathbf{c}(\tau - 1) + \mathbf{\Omega}(\tau - 1)\mathbf{o}(\tau - 1) + \xi_V(\tau)$

    $\mathbf{o}(\tau) = \mathbf{0}$
    $n \leftarrow \arg\max (\mathbf{V} - \mathbf{T}) - \xi_T(\tau))$
    if $V_n > T_n$ **then**
      $o_n(\tau) = 1$
      $\mathbf{F}_n(\tau) = \mathbf{F}_n(\tau - 1) + \epsilon_F(\alpha \mathbf{x}(\tau - 1) - \mathbf{F}_n(\tau - 1))$
      $\mathbf{\Omega}_n(\tau) = \mathbf{\Omega}_n(\tau - 1) - \epsilon_\Omega(\beta(\mathbf{V}(\tau - 1) + \mu \mathbf{r}(\tau - 1)) + \mathbf{\Omega}_n(\tau - 1))$
      $\Omega_{nn}(\tau) = \Omega_{nn}(\tau - 1) - \epsilon_\Omega \mu$

    $\mathbf{r}(\tau) = (1 - \lambda \,\mathrm{dt})\mathbf{r}(\tau - 1) + \mathbf{o}(\tau - 1)$

Algorithm 2 Pseudo-code for simulation of Dales network (Fig. 3)

procedure SIMULATION
  $N_E, N_I, I \leftarrow$ number of cells in E and I populations and input dimensions
  $\lambda, \lambda_F, \lambda_{EI} \leftarrow$ membrane leak and integration time constants for FF rule
  $\mathbf{F}(0) \leftarrow$ initial feedforward weights
  $\mathbf{\Omega}^{EE}(0), \mathbf{\Omega}^{EI}(0), \mathbf{\Omega}^{II}(0), \mathbf{\Omega}^{IE}(0) \leftarrow$ initial recurrent weights
  $S \leftarrow$ total simulation time
  dt $\leftarrow$ time-step
  $\epsilon_F, \epsilon_\Omega \leftarrow$ learning rates
  $\alpha, \beta \leftarrow$ scaling factors in learning equations
  $\mu \leftarrow$ L2 cost
  $T^E, T^I \leftarrow$ threshold
  $\sigma, \eta \leftarrow$ standard deviation of signal, time-scale of smoothing kernel
  $R^E, R^I \leftarrow$ refractory periods of the excitatory and inhibitory neurons
top:
  $\mathbf{V}^E(0), \mathbf{V}^I(0) \leftarrow \mathbf{0}$ (initial voltage)
  $\mathbf{o}^E(0), \mathbf{o}^I(0) \leftarrow \mathbf{0}$ (initial spikes)
  $\mathbf{r}^E(0), \mathbf{r}^I(0) \leftarrow \mathbf{0}$ (initial filtered spikes)
  $\mathbf{R}^E(0), \mathbf{R}^I(0) \leftarrow \mathbf{0}$ (initial refractory periods spikes)
  $\Gamma \leftarrow$ closest integer to S/dt (number of simulation steps)
  $\mathbf{x}(\tau) \leftarrow$ drawn from $\mathcal{N}(0, \sigma^2 \mathbf{I}_I)$ for all $\tau = 1 \dots \Gamma$
  $\mathbf{x}(\tau) \leftarrow \mathbf{x}(\tau)$ filtered with Gaussian kernel of width η over time
  $\xi_E^V(\tau), \xi_I^V(\tau) \leftarrow$ drawn from $\mathcal{N}(0, \sigma^2_{\xi_V} \mathbf{I}_N)$ for all $\tau = 1 \dots \Gamma$
  $\xi_E^T(\tau), \xi_I^T(\tau) \leftarrow$ drawn from $\mathcal{N}(0, \sigma^2_{\xi_T} \mathbf{I}_N)$ for all $\tau = 1 \dots \Gamma$
loop:
  for $\tau = 1$ **to** Γ **do**
    $\mathbf{c}(\tau - 1) = \mathbf{x}(\tau) - \mathbf{x}(\tau - 1) + \lambda \,\mathrm{dt}\, \mathbf{x}(\tau - 1)$
    $\mathbf{c}_E(\tau) = (1 - \lambda_F \mathrm{dt}) \mathbf{c}_E(\tau - 1) + \mathbf{c}(\tau - 1)$
    $\mathbf{c}_I(\tau) = (1 - \lambda_{EI} \mathrm{dt}) \mathbf{c}_E(\tau - 1) + \mathbf{o}_E(\tau - 1)$
    $\mathbf{V}^E(\tau) = (1 - \lambda \,\mathrm{dt}) \mathbf{V}^E(\tau - 1) + \mathrm{dt}\, \mathbf{F}(\tau - 1)^\top \mathbf{c}(\tau - 1) + \mathbf{\Omega}^{EE}(\tau - 1) \mathbf{o}^E(\tau - 1) + \mathbf{\Omega}^{IE}(\tau - 1) \mathbf{o}^I(\tau - 1) + \xi_E^V(\tau)$

    $\mathbf{o}^E(\tau) = \mathbf{0}$
    $n_E \leftarrow \arg\max \left(\mathbf{V}^E - \mathbf{T^E} \right) - \xi_E^T(\tau))$
    if $V^E_{n_E} > T^E_{n_E}$ & $R^E_{n_E}(\tau - 1) < 0$ **then**
      $o^E_{n_E}(\tau) = 1$
      $R^E_{n_E}(\tau - 1) = R^E_{max}$
      $\mathbf{F}_{n_E}(\tau) = \mathbf{F}_{n_E}(\tau - 1) + \epsilon_F (\alpha \mathbf{c}_E(\tau - 1) - \mathbf{F}_{n_E}(\tau - 1))$
      $\mathbf{\Omega}^{EE}_{n_E}(\tau) = \mathbf{\Omega}^{EE}_{n_E}(\tau - 1) - \epsilon_\Omega (\beta (\mathbf{V}^E(\tau - 1) + \mu \mathbf{r}^E(\tau - 1)) + \mathbf{\Omega}^{EE}_{n_E}(\tau - 1))$
      $\Omega^{EE}_{n_E n_E}(\tau) = \Omega {n_E}{n_E}^{EE}(\tau - 1) - \epsilon_\Omega \mu$
      $\mathbf{\Omega}^{EI}_{n_E}(\tau) = \mathbf{\Omega}^{EI}_{n_E}(\tau - 1) - \epsilon_\Omega (\beta (\mathbf{V}^I(\tau - 1) + \mu \mathbf{r}^I(\tau - 1)) + \mathbf{\Omega}^{EI}_{n_E}(\tau - 1))$

    $R^E(\tau) = R^E(\tau - 1) - 1$
    $\mathbf{r}^E(\tau) = (1 - \lambda \,\mathrm{dt}) \mathbf{r}^E(\tau - 1) + \mathbf{o}^E(\tau - 1)$

$\mathbf{V}^I(\tau) = (1-\lambda\,\mathrm{dt})\mathbf{V}^I(\tau-1) + \mathbf{\Omega}^{EI}(\tau)\mathbf{o}^E(\tau) + \mathbf{\Omega}^{II}(\tau-1)\mathbf{o}^I(\tau-1) + \xi_I^V(\tau)$

$\mathbf{o}^I(\tau) = \mathbf{0}$

$n_I \leftarrow \arg\max\left(\mathbf{V}^I - \mathbf{T^I}\right) - \xi_I^T(\tau))$

if $V^I_{n_I} > T^I_{n_I}$ & $R^I_{n_I}(\tau-1) < 0$ **then**

$o^I_{n_I}(\tau) = 1$

$R^I_{n_I}(\tau-1) = R^I_{max}$

$\mathbf{\Omega}^{EI}_{n_I}(\tau) = \mathbf{\Omega}^{EI}_{n_I}(\tau-1) + \epsilon_F(\alpha \mathbf{c}_I(\tau-1) - \mathbf{\Omega}^{EI}_{n_I}(\tau-1))$

$\mathbf{\Omega}^{II}_{n_I}(\tau) = \mathbf{\Omega}^{II}_{n_I}(\tau-1) - \epsilon_\Omega(\beta(\mathbf{V}^I(\tau-1) + \mu\mathbf{r}^I(\tau-1)) + \mathbf{\Omega}^{II}_{n_I}(\tau-1))$

$\Omega^{II}_{n_I n_I}(\tau) = \Omega {n_I}{n_I}^{II}(\tau-1) - \epsilon_\Omega\mu$

$\mathbf{\Omega}^{IE}_{n_I}(\tau) = \mathbf{\Omega}^{IE}_{n_I}(\tau-1) - \epsilon_\Omega(\beta(\mathbf{V}^E(\tau-1) + \mu\mathbf{r}^E(\tau-1)) + \mathbf{\Omega}^{IE}_{n_I}(\tau-1))$

$R^I(\tau) = R^I(\tau-1) - 1$

$\mathbf{r}^I(\tau) = (1-\lambda\,\mathrm{dt})\mathbf{r}^I(\tau-1) + \mathbf{o}^I(\tau-1)$

Parameters	Figure 2	Figure 3	Figure 4
Number of neurons N	20	$N_E = 300/60$ (BC/D) $N_I = 75/15$ (BC/D)	64
Dimension of input I	2	3	25
Time step dt	10^{-3}s	10^{-4}s	$6.25 \cdot 10^{-5}$s
Membrane leak λ	50 s^{-1}	50 s^{-1}	8 s^{-1}
Integration time constant, FF rule	$\lambda_F = \lambda$	$\lambda_F = 6\lambda$ $\lambda_{E-I} = \lambda$	$\lambda_F = 125\lambda$
Standard deviation of input σ	$2 \cdot 10^3$	$2 \cdot 10^3$	-
Time scale of input kernel η	6 ms	6 ms	-
Threshold T	0.5	0.5	dynamic
standard dev. of voltage noise σ_{ξ_V}	10^{-3}	10^{-3}	0
standard dev. of threshold noise σ_{ξ_T}	$2 \cdot 10^{-2}$	$2 \cdot 10^{-2}$	$5 \cdot 10^{-3}$
Learning rate ϵ_Ω	10^{-4}	10^{-4}	variable
Learning rate ϵ_F	10^{-5}	10^{-5}	variable
Scaling factor α	0.21	0.21	1
Scaling factor β	1.25	1	1
L2 cost μ	0.02	$\mu_E = 0.02, \mu_I = 0$	0.1
Initial scale γ	0.8	1	-
Initial scale ω	-0.5	$\omega_{EE} = -0.02$ $\omega_{II} = -0.5$ $\omega_{EI} = 0.5$ $\omega_{IE} = -0.3$	-

Table 1: Simulation parameters for all simulations.

6.3 Tuning Curves

To compute the tuning curves (Figures 2 and 3), we define a circle in a 2D plane and then sample inputs uniformly on this circle. For each trial a constant input ('orientation') is sampled from this circle and presented to the network (running without plasticity). At the end of the trial, the average firing rate of each neuron is computed over all the duration of the presentation of the input except for the initial transient period.

6.4 Fano Factor and Coefficient of Variation

The Fano factor and the coefficient of variation are computed as follows. A random direction is chosen in the input space and presented multiple times to the network (running without plasticity). For each trial the spike count c of the neurons is computed. then we compute the the Fano factor for each neuron using the formula :

$$F_n = \frac{\sigma^2_{c(n)}}{\mu_{c(n)}}$$

$\sigma^2_{c(n)}$ and $\mu_{c(n)}$ are respectively the standard deviation and the mean of the the spike count of neuron n. This procedure is repeated using different random input directions. The final Fano factor is an average over the input directions and the neurons in the population.
The Coefficient of variation (CV) is computed using the same inputs. For each trial, instead of the spike count, we pool the interspike intervals (ISI) of all neurons. The formula used to compute the CV is

$$CV = \frac{\sigma_{ISI}}{\mu_{ISI}}$$

As for the Fano factor, the final CV is an average over the different input directions.

6.5 Simulation of speech signal learning

To learn the speech signals (Figure 4), slight modifications were added to the previous simulation scheme. In a non-whitened scenario, partial learning of the inhibitory recurrent connections can result in a large proportion of completely silent neurons. Since plasticity requires both pre- and post-synaptic spiking, these neuron never 'recover' or participate in the representation. To avoid this issue, we used a dynamic threshold that decreases for unresponsive neurons and increases for neurons that are too active. The threshold decreased by $-\epsilon_F$ for a neuron that did not fire any spike in a sliding window of 2.5s, and increased by ϵ_F if its firing rate exceeded 20Hz in the last 2.5s. After about 1000 iterations, the firing rates are always maintained between these two bounds and the thresholds remain constant for the rest of the learning.

In Figure 4, the initial recurrent and feedforward weights are drawn from a normal distribution with standard deviations of 0.1 and 0.02 for the feedforward and the recurrent weights respectively; These initial weights are not normalized. The diagonal elements of the recurrent connectivity matrix (the resets) are equal to -0.8. Such strong inhibitory autapses insure the stability of the network in the initial state. In order to speed up learning, we used initially large learning

rates ($\epsilon_\Omega = 10^{-2}$, $\epsilon_F = 10^{-3}$) that were progressively decreased to $\epsilon_\Omega = 10^{-4}$ and $\epsilon_F = 10^{-5}$. For re-training to the new non-speech stimulus, we used the learning rates $\epsilon_\Omega = 10\epsilon_F = 10^{-2}$). For re-training the feed-forward connections without the lateral connections, we used $\epsilon_F = 0.2510^{-2}$).

7 Learning a Speech signal, Supplementary results

This section provides a figure with additional results for the network trained with speech signals (Figure 4 in the main paper). Here, we report the initial, learnt, and re-trained input and decoding weights. We also show the result of re-learning a new sound when training the feed-forward, but not the recurrent connections.

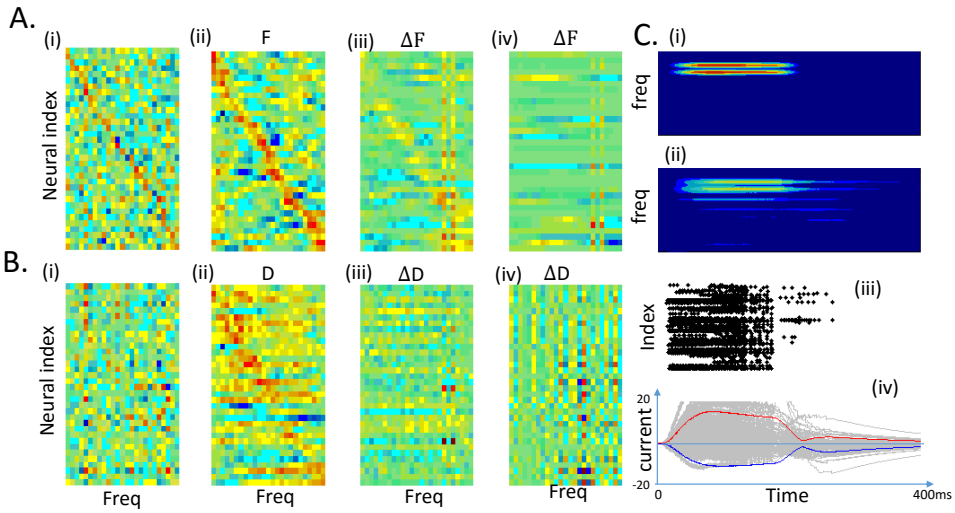

Figure S2: Supplementary result for learning speech sounds. A. (i) Feed-forward weights (similar to "receptive fields") of the neurons. The initial weights are random. The diagonal appears because these were sorted according to maximal frequency. Bluish colors correspond to negative values, reddish colors to positive values. (ii) After learning, the receptive fields have an excitatory sub-field (reddish colors), and most neurons also exhibit one or two strong inhibitory subfields (bluish colors). These observations are broadly compatible with receptive fields observed in the mammalian auditory pathway, and notably in the representation of speech signals in A1 [13]. Note that the weights have been re-sorted according to maximal frequency. (iii) After re-training with the new stimulus (see panel C(i)) the receptive fields change selectively (positively and negatively) at the position of the trained frequencies. The panel represents the difference between the weights after and before the retraining. These frequency-selective changes in receptive field structure is in line with fast plastic changes of receptive fields observed following behavioral training [12]. There is also a small decrease in gain at other frequencies, due to the competition with the new stimulus. (iv) When blocking the learning of the lateral connections prior to the introduction of the new stimulus, so that only the feed-forward connections can be learnt, the receptive fields change in a similar fashion for the trained frequencies (without change in gain). B. Same as in A, but for the decoding weights. (i) Decoding weights before learning appear random. Note that they are sorted as in A(i) to allow comparison of feedforward and decoding weights for each neuron. (ii) After learning, the decoding weights are more structured and broader than the feedforward weights. This is compatible to the decoding filter of speech measured in auditory cortex [13]. They have been sorted as in A(ii). (iii) After re-training to the new stimulus, a small number of decoding filters (neurons) "specialize" for the new stimulus, while the other decoding weights change only mildly. This allows the network to minimize its firing rate response to the new stimulus, while still providing an accurate representation of it. (iv) After training only the feed-forward connections, the changes in the decoder are massive and disorganized. This reflects a severe degradation in coding performance. C. Response of the network after re-training with the new stimulus, but only the feedforward weights, not the recurrent weights. (i) The new stimulus. (ii) The estimate of the stimulus after re-training is poorer than it was before (see Figure 4 in main text). (iii) The firing rates have massively increased. (iv) The balance between excitation and inhibition has worsened. Thus, we predict that specifically blocking inhibitory plasticity during exposure to a new stimulus would actually degrade learning performance at the same time that it degrades the EI balance. Note that to avoid a total failure of the network (whose firing rates eventually explodes without training the lateral connections), we divided the learning rate of the feed-forward connections by 4.

References

[1] Martin Boerlin, Christian K. Machens, Sophie Denève. *Predictive Coding of Dynamical Variables in Balanced Spiking Networks.* PLoS Comput Biol 9(11) (2013): e1003258.

[2] Ralph Bourdoukan, David G.T. Barrett, Christian K. Machens, and Sophie Denève. *Learning Optimal Spike-based Representations.* Advances in Neural Information Processing Systems 25 (2012).

[3] Peter Dayan and Larry F. Abbott. *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems.* MIT Press (2005).

[4] Christopher J. Rozell, Don H. Johnson, Richard G. Baraniuk, and Bruno A. Olshausen. *Sparse coding via thresholding and local competition in neural circuits.* Neural Computation 20.10 (2008): 2526–2563

[5] Joel Zylberberg, Jason T. Murphy, and Michael R. DeWeese. *A sparse coding model with synaptically local plasticity and spiking neurons can account for the diverse shapes of V1 simple cell receptive fields.* PLoS Comput Biol 7.10 (2011): e1002250.

[6] Pietro Vertechi, Wieland Brendel, and Christian K. Machens. *Unsupervised learning of an efficient short-term memory network.* Advances in Neural Information Processing Systems 27 (2014).

[7] John J. Hopfield. *Neural networks and physical systems with emergent collective computational abilities.* Proceedings of the National Academy of Sciences, 79.8 (1982): 2554–2558.

[8] John Hertz, Richard G. Palmer, and Anders S. Krogh. *Introduction to the Theory of Neural Computation.* Perseus Publishing, 1991

[9] Barak A. Pearlmutter. *Learning state space trajectories in recurrent neural networks.* Neural Computation, 1.2 (1989): 263–269.

[10] David Susillo and Larry F. Abbott. *Generating coherent patterns of activity from chaotic neural networks.* Neuron, 63.4 (2009): 544–557.

[11] Tim P. Vogels, Henning Sprekeler, Friedmann Zenke, Claudia Clopath, and Wulfram Gerstner. *Inhibitory plasticity balances excitation and inhibition in sensory pathways and memory networks.* Science, 334.6062 (2011):1569–1573.

[12] Yin P, Fritz JB, Shamma SA. *Rapid spectrotemporal plasticity in primary auditory cortex during behavior.* J Neurosci. 2014 Mar 19;34(12):4396-408.

[13] Mesgarani N, David SV, Fritz JB, Shamma SA. *Mechanisms of noise robust representation of speech in primary auditory cortex.* Proc Natl Acad Sci U S A. 2014 May 6;111(18):6792-7.

[14] Pengsheng Zheng, Christos Dimitrakakis, and Jochen Triesch. *Network Self-Organization Explains the Statistics and Dynamics of Synaptic Connection Strengths in Cortex.* PLoS Comput Biol 9(01) (2013): e1002848

[15] Andreea Lazar, Gordon Pipa, and Jochen Triesch. *SORN: A Self-Organizing Recurrent Neural Network.* Frontiers in Computational Neuroscience 3 (2009): 23

[16] Wieland Brendel. *On the Structure and self-organized Formation of Neural Population Responses.* PhD Thesis, Ecole Normale Supérieure Paris

[17] Aapo Hyvärinen, Juha Karhunen, and Erkki Oja. *Independent component analysis.* Wiley & Sons (2001).

[18] Bruno A. Olshausen and David J. Field. *Sparse coding with an overcomplete basis set: A strategy employed by V1?* Vision Research 37.23 (1997):3311–3325.

[19] Anthony J. Bell and Terrence J. Sejnowski. *The independent components of natural scenes are edge filters.* Vision Research 37.23 (1997): 3327–3338.

[20] Ganguli Deep and Simoncelli P. Eero *Efficient sensory encoding and Bayesian inference with heterogeneous neural populations.* Neural computation (2014).

Chapter 4

Learning a Linear Dynamical System

We have previously considered the problem of coding or how recurrent networks of integrate-and-fire neurons can learn to optimally represent their inputs. In the following paper, we extend the framework to the learning of more complex computations. Here networks do not learn to solely represent their inputs but operate transformations on them. In the auto-encoder, the desired output is fed as an input to the network. Here the input of the network $\mathbf{c}$ and the desired output $\mathbf{x}$ are related through a linear differential equation:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{c}(t) \tag{4.1}$$

where $\mathbf{A}$ is the state matrix of the system.

(Boerlin et al., 2013) derived a recurrent network of spiking neurons that implements such a dynamical system. Here we briefly expose their derivation. This derivation is also illustrated in less detail in the second section of the following paper (Bourdoukan and Denève, 2015).

Decoding Scheme

The decoding scheme is the same as the one used in the auto-encoder. The spike trains are modeled as a sum of Dirac functions $o_i(t) = \sum_k \delta(t - t_i^k)$ where o_i is the spike train of neuron i and t_j^k are the times of its spikes. We assume that the variables can be decoded linearly by a leaky integration of the spike trains as follows:

$$\dot{\hat{x}}_j(t) = -\lambda \hat{x}_j(t) + \sum_{n=1}^{N} D_{jn} o_n(t) \tag{4.2}$$

where $\hat{x}_j$ is the j^{th} decoded variable, λ is the leak of the decoder and D_{jn} is the decoding weight from neuron n to the j_{th} component of the decoded

variable. This decoding scheme is akin to a synaptic integration of spikes in a postsynaptic neuron. In fact, every spike emitted by neuron i adds an exponential kernel of amplitude D_{nj} to the j^{th} output unit $\hat{x}_j$, similar to a postsynaptic potential (PSP) provoked by a spike in a postsynaptic neuron. This means that the variables can be decoded by any downstream neuron if it has the right set of synaptic weights. If we define the filtered spike trains r_n:

$$\dot{r}_n(t) = -\lambda r_n(t) + o_n(t) \tag{4.3}$$

Alternatively, we can write the estimate as a function of the these quantities:

$$\hat{x}_j(t) = \sum_{n=1}^{N} D_{jn} r_n(t) \tag{4.4}$$

or in matrix form

$$\hat{\mathbf{x}}(t) = \mathbf{D}\mathbf{r}(t) \tag{4.5}$$

where $\mathbf{D} = (D_{jn})_{1 \leqslant j \leqslant J,\ 1 \leqslant n \leqslant N}$ is the decoding weight matrix and $\mathbf{r} = (r_n)_{1 \leqslant n \leqslant N}$ is the population vector of the filtered spike trains. The filtered spike train r_n can be interpreted as an estimate of the rate of the neuron n up to the factor λ.

Spiking Dynamics

The objective is to minimize the error between the desired output $\mathbf{x}$ and the decoded output from the network $\hat{\mathbf{x}}$. To derive the spiking policy that minimizes the error, we use the same loss function as for the auto-encoder. This loss function contains three terms. One measuring the loss between $\mathbf{x}$ and $\hat{\mathbf{x}}$, a quadratic cost term and a linear cost term that regularize spiking activity:

$$L(t) = \sum_{j=1}^{J} (x_j(t) - \hat{x}_j(t))^2 + \nu \sum_{l=1}^{N} r_l(t) + \mu \sum_{l=1}^{N} r_l^2(t) \tag{4.6}$$

To optimize the spike times in the network, the neurons fire according to a greedy rule that minimizes the previous loss function. Thus, a spike is emitted only if it immediately decreases the loss function L:

$$L(t \mid \text{if neuron } n \text{ spikes}) < L(t \mid \text{if neuron } n \text{ does not spike}) \tag{4.7}$$

Then, following the same steps as for the auto-encoder (Chapter 3, supplementary information), one gets the following spiking condition:

$$\mathbf{D}_n^T(\mathbf{x} - \hat{\mathbf{x}}) - \mu r_n > \frac{\| \mathbf{D}_n \|^2 + \mu + \nu}{2} \tag{4.8}$$

The left side of the inequality is defined as the voltage of neuron n and the right side as its spiking threshold:

$$V_n = \mathbf{D}_n^T(\mathbf{x} - \hat{\mathbf{x}}) - \mu r_n$$
$$T_n = \frac{\| \mathbf{D}_n \|^2 + \mu + \nu}{2}$$

The spiking rule we obtain is very simple: a neuron's spike decreases the error only if the projection of the global error between the target and the estimate onto the neuron's decoding weight vector is larger than half the length of this decoding weight and the cost terms.

Leaky integrate-and-fire Dynamics

by deriving the voltage equation above with respect to time we obtain:

$$\dot{V}_n = \mathbf{D}_n^T(\dot{\mathbf{x}} - \dot{\hat{\mathbf{x}}}) - \mu \dot{r}_n \tag{4.9}$$

Until now the derivation is exactly the same as for an auto-encoder. However, since $\mathbf{x}$ now has specific linear dynamics, when we replace $\dot{\mathbf{x}}$, $\dot{\hat{\mathbf{x}}}$ and $\dot{r}_n$ by their expression following 4.1, 4.2 and 4.3, we obtain :

$$\dot{V}_n = \mathbf{D}_n^T(\mathbf{A}\mathbf{x} + \mathbf{c} + \lambda\hat{\mathbf{x}} - \mathbf{D}\mathbf{o}) + \mu\lambda r_n - \mu o_n \tag{4.10}$$

.

Then, by adding and subtracting the term $\lambda\mathbf{x}$ to the previous equation

$$\begin{aligned}
\dot{V}_n &= \mathbf{D}_n^T(\mathbf{A}\mathbf{x} + \mathbf{c} + \lambda\mathbf{x} - \lambda\mathbf{x} + \lambda\hat{\mathbf{x}} - \mathbf{D}\mathbf{o}) + \mu\lambda r_n - \mu o_n \\
&= \mathbf{D}_n^T(\mathbf{A}\mathbf{x} + \mathbf{c} + \lambda\mathbf{x} - \lambda(\mathbf{x} - \hat{\mathbf{x}}) - \mathbf{D}\mathbf{o}) + \mu\lambda r_n - \mu o_n \\
&= \lambda\mathbf{D}_n^T(\mathbf{x} - \hat{\mathbf{x}}) - \mu\lambda r_n + \mathbf{D}_n^T(\mathbf{A}\mathbf{x} + \mathbf{c} + \lambda\mathbf{x} - \mathbf{D}\mathbf{o}) + \mu\lambda r_n - \mu o_n \\
&= -\lambda(\mathbf{D}_n^T(\mathbf{x} - \hat{\mathbf{x}}) - \mu r_n) + \mathbf{D}_n^T(\mathbf{A}\mathbf{x} + \mathbf{c} + \lambda\mathbf{x} - \mathbf{D}\mathbf{o}) - \mu o_n \\
&= -\lambda V_n + \mathbf{D}_n^T(\mathbf{A}\mathbf{x} + \mathbf{c} + \lambda\mathbf{x} - \mathbf{D}\mathbf{o}) - \mu o_n
\end{aligned}$$

We assume that the network output is tracking $\mathbf{x}$ accurately at a time t such that $\hat{\mathbf{x}} \approx \mathbf{x}$ and using this auto-consistency argument we replace $\mathbf{x}$ by $\hat{\mathbf{x}}$ in the previous equation and obtain:

$$\begin{aligned}
\dot{V}_n &= -\lambda V_n + \mathbf{D}_n^T(\mathbf{A}\hat{\mathbf{x}} + \mathbf{c} + \lambda\hat{\mathbf{x}} + \mathbf{D}\mathbf{o}) - \mu o_n \\
&= -\lambda V_n + \mathbf{D}_n^T\mathbf{c} + \mathbf{D}_n^T(\mathbf{A} + \lambda\mathbf{I})\hat{\mathbf{x}} - \mathbf{D}_n^T\mathbf{D}\mathbf{o} - \mu o_n
\end{aligned}$$

In the last equation, we replace $\hat{\mathbf{x}}$ according to equation 4.5

$$\dot{V}_n = -\lambda V_n + \mathbf{D}_n^T\mathbf{c} + \mathbf{D}_n^T(\mathbf{A} + \lambda\mathbf{I})\mathbf{D}\mathbf{r} - \mathbf{D}_n^T\mathbf{D}\mathbf{o} - \mu o_n \tag{4.11}$$

Which we rewrite in a full Matrix form :

$$\dot{\mathbf{V}} = -\lambda\mathbf{V} + \mathbf{D}^T\mathbf{c} + \mathbf{D}^T(\mathbf{A} + \lambda\mathbf{I})\mathbf{D}\mathbf{r} - (\mathbf{D}^T\mathbf{D} + \mu\mathbf{I})\mathbf{o} \tag{4.12}$$

where $\mathbf{V}$ is the vector of the membrane potentials of all the neurons. Indeed, these are the dynamic of a recurrently connected network of leaky integrate-and-fire neurons. In addition to the feedforward command $\mathbf{c}$, the neurons receive two different recurrent inputs. A fast input consisting of raw spikes with weights $\mathbf{W}^f = (\mathbf{D}^T\mathbf{D} + \mu\mathbf{I})$ identical to the recurrent weights of the auto-encoder. We refer to these weights as fast. The fast recurrent input has the same function as in the auto-encoder, as it insures that the representation of the output is shared among neurons. In addition to this fast input, the neurons receive a slower recurrent input consisting of the filtered spike trains $\mathbf{r}$, which are transmitted with what we define as the slow weights $\mathbf{W}^s = \mathbf{D}^T(\mathbf{A} + \lambda\mathbf{I})\mathbf{D}$. These inputs are responsible for the generation of the right temporal dynamics of the output. They directly depend on the underlying linear dynamical system through the matrix $\mathbf{A}$.

Learning the recurrent connectivities

In the previous chapter, we have derived a rule for learning fast connectivity. The novelty of this work consists in deriving a learning rule for the slow connectivity. The learning of such connectivity uses a supervising error signal that is fed back into the neurons. Indeed, the fast and slow weights learn simultaneously with these local plasticity rules. In this framework, the encoding weights are not subject to learning but are fixed to their optimal values.

The paper is organized as follows: in the first section we briefly describe, as a reminder, the learning of the fast connections. Then we show schematically how, starting from an auto-encoder, the network is able to implement a linear dynamical system if slow connections are added. Finally, we derive the learning rule of the slow weights and show its effectiveness through simulations.

Enforcing balance allows local supervised learning in spiking recurrent networks

Ralph Bourdoukan
Group For Neural Theory, ENS Paris
Rue dUlm, 29, Paris, France
ralph.bourdoukan@gmail.com

Sophie Deneve
Group For Neural Theory, ENS Paris
Rue dUlm, 29, Paris, France
sophie.deneve@ens.fr

Abstract

To predict sensory inputs or control motor trajectories, the brain must constantly learn temporal dynamics based on error feedback. However, it remains unclear how such supervised learning is implemented in biological neural networks. Learning in recurrent spiking networks is notoriously difficult because local changes in connectivity may have an unpredictable effect on the global dynamics. The most commonly used learning rules, such as temporal back-propagation, are not local and thus not biologically plausible. Furthermore, reproducing the Poisson-like statistics of neural responses requires the use of networks with balanced excitation and inhibition. Such balance is easily destroyed during learning. Using a top-down approach, we show how networks of integrate-and-fire neurons can learn arbitrary linear dynamical systems by feeding back their error as a feed-forward input. The network uses two types of recurrent connections: fast and slow. The fast connections learn to balance excitation and inhibition using a voltage-based plasticity rule. The slow connections are trained to minimize the error feedback using a current-based Hebbian learning rule. Importantly, the balance maintained by fast connections is crucial to ensure that global error signals are available locally in each neuron, in turn resulting in a local learning rule for the slow connections. This demonstrates that spiking networks can learn complex dynamics using purely local learning rules, using E/I balance as the key rather than an additional constraint. The resulting network implements a given function within the predictive coding scheme, with minimal dimensions and activity.

The brain constantly predicts relevant sensory inputs or motor trajectories. For example, there is evidence that neural circuits mimic the dynamics of motor effectors using internal models [1]. If the dynamics of the predicted sensory and motor variables change in time, these models may become false [2] and therefore need to be readjusted through learning based on error feedback.

From a modeling perspective, supervised learning in recurrent networks faces many challenges. Earlier models have succeeded in learning useful functions at the cost of non local learning rules that are biologically implausible [3, 4]. More recent models based on reservoir computing [5–7] transfer the learning from the recurrent network (with now "random", fixed weights) to the readout weights. Using this simple scheme, the network can learn to generate complex patterns. However, the majority of these models use abstract rate units and are yet to be translated into more realistic spiking networks. Moreover, to provide a sufficiently large reservoir, the recurrent network needs to be large, balanced and have a rich and high dimensional dynamics. This typically generates far more activity than strictly required, a redundancy that can be seen as inefficient.

On the other hand, supervised learning models involving spiking neurons have essentially concentrated on the learning of precise spike sequences [8–10]. With some exceptions [10,11] these models use feed-forward architectures [12]. In a balanced recurrent network with asynchronous, irregular and highly variable spike trains, such as those found in cortex, the activity has been shown to be

chaotic [13, 14]. This leads to spike timing being intrinsically unreliable, rendering a representation of the trajectory by precise spike sequences problematic. Moreover, many configurations of spike times may achieve the same goal [15].

Here we derive two local learning rules that drive a network of leaky integrate-and-fire (LIF) neurons into implementing a desired linear dynamical system. The network is trained to minimize the objective $\|\mathbf{x}(t) - \hat{\mathbf{x}}(t)\|^2 + H(\mathbf{r})$, Where $\hat{\mathbf{x}}(t)$ is the output of the network decoded from the spikes, $\mathbf{x}(t)$ is the desired output, and $H(\mathbf{r})$ is a cost associated with firing (penalizing unnecessary activity, and thus enforcing efficiency). The dynamical system is linear, $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{c}$, with $\mathbf{A}$ being a constant matrix and $\mathbf{c}$ a time varying command signal. We first study the learning of an autoencoder, i.e., a network where the desired output is fed to the network as a feedforward input. The autoencoder learns to represent its inputs as precisely as possible in an unsupervised fashion. After learning, each unit represents the encoding error made by the entire network. We then show that the network can learn more complex computations if slower recurrent connections are added to the autoencoder. Thus, it receives the command $\mathbf{c}$ along with an error signal and learns to generate the output $\hat{\mathbf{x}}$ with the desired temporal dynamics. Despite the spike-based nature of the representation and of the plasticity rules, the learning does not enforce precise spike timing trajectories but, on the contrary, enforces irregular and highly variable spike trains.

1 Learning a balance : global becomes local

Using a predictive coding strategy [15–17], we build a network that learns to accurately represent its inputs while expending the least amount of spikes. To introduce the learning rules and explain how they work, we start by describing the optimized network (after learning).

Let us first consider a set of unconnected integrate-and-fire neurons receiving shared input signals $\mathbf{x} = (x_i)$ through feedforward connections $\mathbf{F} = (F_{ji})$. We assume that the network performs predictive coding, i.e. it subtracts from each of these input signals an estimate $\hat{\mathbf{x}}$ obtained by decoding the output spike trains (fig 1A). Specifically, $\hat{x}_i = \sum D_{ij} r_j$, where $\mathbf{D} = (D_{ij})$ are the decoding weights and $\mathbf{r} = (r_j)$ are the filtered spike trains which obey $\dot{r}_j = -\lambda r_j + o_j$ with $o_j(t) = \sum_k \delta(t - t_j^k)$ being the spike train of neuron j and t_j^k are the times of its spikes. Note that such an autoencoder automatically maintains an accurate representation, because it responds to any encoding error larger than the firing threshold by increasing its response and in turn decreasing the error. It is also efficient, because neurons respond only when input and decoded signals differ. The autoencoder can be equivalently implemented by lateral connections, rather than feedback targeting the inputs (Fig 1A). These lateral connections combine the feedforward connections and the decoding weights and they subtract from the feedforward inputs received by each neuron. The membrane potential dynamics in this recurrent network are described by:

$$\dot{\mathbf{V}} = -\lambda \mathbf{V} + \mathbf{F}\mathbf{s} + \mathbf{W}\mathbf{o} \tag{1}$$

where $\mathbf{V}$ is the vector of the membrane potentials of the population, $\mathbf{s} = \dot{\mathbf{x}} + \lambda \mathbf{x}$ is the effective input to the population, $\mathbf{W} = -\mathbf{F}\mathbf{D}$ is the connectivity matrix, and $\mathbf{o}$ is the population vector of the spikes. Neuron i has threshold $T_i = \|\mathbf{F}_i\|^2/2$ [15]. When input channels are independent and the feed-forward weights are distributed uniformly on a sphere then the optimal decoding weights $\mathbf{D}$ are equal to the encoding weights $\mathbf{F}$ and hence the optimal recurrent connectivity $\mathbf{W} = -\mathbf{F}\mathbf{F}^T$ [17]. In the following we assume that this is always the case and we choose the feedforward weights accordingly.

In this auto-encoding scheme having a precise representation of the inputs is equivalent to maintaining a precise balance between excitation and inhibition. In fact, the membrane potential of a neuron is the projection of the global error of the network on the neurons's feedforward weight ($V_i = \mathbf{F}_i(\mathbf{x} - \hat{\mathbf{x}})$ [15]). If the output of the network matches the input, the recurrent term in the membrane potential, $\mathbf{F}_i\hat{\mathbf{x}}$, should precisely cancel the feedforward term $\mathbf{F}_i\mathbf{x}$. Therefore, in order to learn the connectivity matrix $\mathbf{W}$, we tackle the problem through balance, which is its physiological characterization. The learning rule that we derive achieves efficient coding by enforcing a precise balance at a single neuron level. It makes the network converge to a state where each presynaptic spike cancels the recent charge that was accumulated by the postsynaptic neuron (Fig 1B). This accumulation of charge is naturally represented by the postsynaptic membrane potential V_i, which jumps upon the arrival of a presynaptic spike by a magnitude given by the recurrent weight W_{ij} due

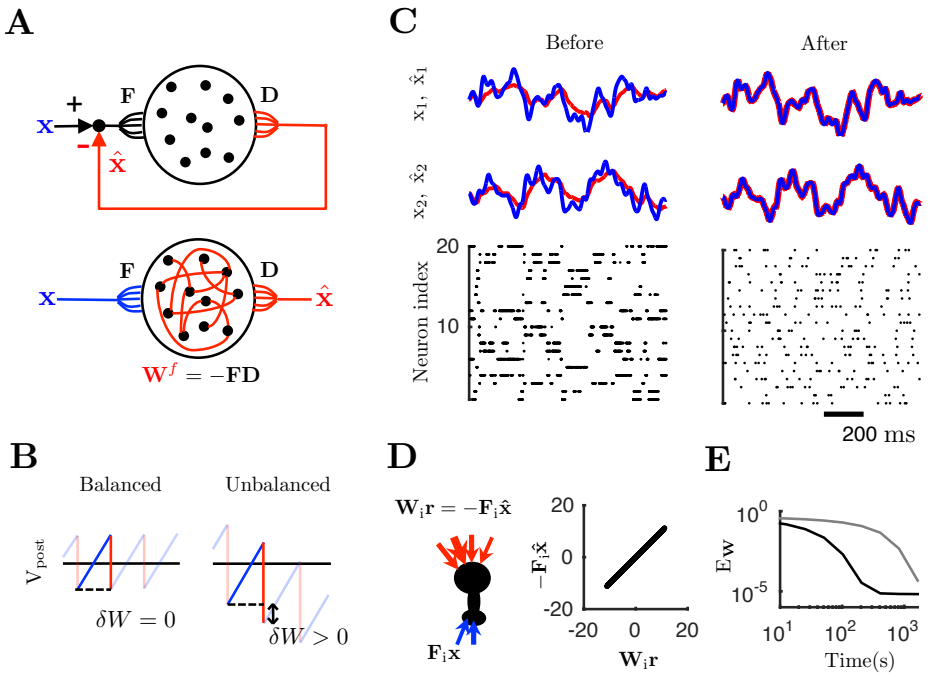

Figure 1: **A**: a network preforming predictive coding. Top panel: a set of unconnected leaky integrate-and-fire neurons receiving the error between a signal and their own decoded spike trains. Bottom panel: the previous architecture is equivalent to the recurrent network with lateral connections equal to the product of the encoding and the decoding weights. **B**: illustration of the learning of an inhibitory weight. The trace of the membrane potential of a postsynaptic neuron is shown in blue and red. The blue lines correspond to changes due to the integration of the feedforward input, and the red to changes caused by the integration of spikes from neurons in the population. The black line represents the resting potential of the neuron. In the left panel the presynaptic spike perfectly cancels the accumulated feedforward current during a cycle and therefore there is no learning. In the right panel the inhibitory weight is too strong and thus creates imbalance in the membrane potential. Therefore, it is depressed by learning. **C**: learning in a 20-neuron network. Top panels: the two dimensions of the input (blue lines) and the output (red lines) before (left) and after (right) learning. Bottom panels: raster plots of the spikes in the population. **D**, left panel: after learning each neuron receives a local estimate of the output of the network through lateral connections (red arrows). Right panel: scatter plot of the output of the network projected on the feedforward weights of the neurons versus the recurrent input they receive. **E**: the evolution of the mean error between the recurrent weights of the network and the optimal recurrent weights $-\mathbf{F}\mathbf{F}^T$ using the rule defined by equation 2 (black line) and the rule in [16] (gray line). Note that our rule is different from [16] because it operates on a finer time-scale and reaches the optimal balanced state with more than one order of magnitude faster. This speed-up is important because, as we will see below, some computations require a very fast restoration of this balance.

to the instantaneous nature of recurrent synapses. Because the two charges should cancel each other, the greedy learning rule is proportional to the sum of both quantities:

$$\delta W_{ij} \propto -(V_i + \beta W_{ij}) \tag{2}$$

where V_i is the membrane potential of the postsynaptic neuron, W_{ij} is the recurrent weight from neuron j to neuron i, and the factor β controls the overall magnitude of lateral weights. More importantly, β regularizes the cost penalizing the total spike count in the population (i.e. $H(\mathbf{r}) = \mu \sum_i r_i$ where μ is the effective linear cost [15]). The example of an inhibitory synapse $W_{ij} < 0$ is illustrated in figure 1B. If neuron i is too hyperpolarized upon the arrival of a presynaptic spike from neuron j, i.e., if the inhibitory weight W_{ij} is smaller than $-V_i/\beta$, the absolute weight of

the synapse (the amplitude of the IPSP) is decreased. The opposite occurs if the membrane is too depolarized. The synaptic weights thus converge when the two quantities balance each other on average $W_{ij} = -\langle V_i \rangle_{t_j}/\beta$, where t_j are the spike times of the presynaptic neuron j.

Fig 1C shows the learning in a 20-neuron network receiving random input signals. For illustration purposes the weights are initialized with very small values. Before learning, the lack of lateral connectivity causes neurons to fire synchronously and regularly. After learning, spike trains are sparse, irregular and asynchronous, despite the quasi absence of noise in the network. Even though the firing rates decrease globally, the quality of the input representation drastically improves over the course of learning. Moreover, the convergence of recurrent weights to their optimal values is typically quick and monotonic (Fig 1E).

By enforcing balance, the learning rule establishes an efficient and reliable communication between neurons. Because $\mathbf{V} = \mathbf{F}\mathbf{x} - \mathbf{F}\mathbf{F}^T\mathbf{r} = \mathbf{F}(\mathbf{x} - \hat{\mathbf{x}})$, every neuron has access - through its recurrent input - to the network's global coding error projected on its feedforward weight (Fig 1D). This local representation of the network's global performance is crucial in the supervised learning scheme we describe in the following sections.

2 Generating temporal dynamics within the network

While in the previous section we presented a novel rule that drives a spiking network into efficiently representing its inputs, we are generally interested in networks that perform more complex computations. It has been shown already that a network having two synaptic time scales can implement an arbitrary linear dynamical system [15]. We briefly summarize this approach in this section.

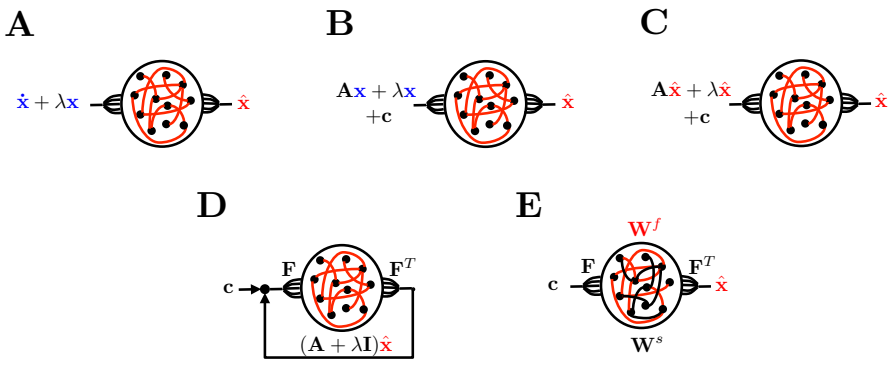

Figure 2: The construction of a recurrent network that implements a linear dynamical system.

In the autoencoder presented above, the effective input to the network is $\mathbf{s} = \dot{\mathbf{x}} + \lambda\mathbf{x}$ (Fig 2A). We assume that $\mathbf{x}$ follows linear dynamics $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{c}$, where $\mathbf{A}$ is a constant matrix and $\mathbf{c}(t)$ is a time varying command. Thus, the input can be expanded to $\mathbf{s} = \mathbf{A}\mathbf{x} + \mathbf{c} + \lambda\mathbf{x} = (\mathbf{A} + \lambda\mathbf{I})\mathbf{x} + \mathbf{c}$ (Fig 2B). Because the output of the network $\hat{\mathbf{x}}$ approximates $\mathbf{x}$ very precisely, they can be interchanged. According to this self-consistency argument, the external input term $(\mathbf{A} + \lambda\mathbf{I})\mathbf{x}$ is replaced by $(\mathbf{A} + \lambda\mathbf{I})\hat{\mathbf{x}}$ which only depends on the activity of the network (Fig 2C). This replacement amounts to including a global loop that adds the term $(\mathbf{A} + \lambda\mathbf{I})\hat{\mathbf{x}}$ to the source input (Fig 2D). As in the autoencoder, this can be achieved using recurrent connections in the form of $\mathbf{F}(\mathbf{A} + \lambda\mathbf{I})\mathbf{F}^T$ (Fig 2E). Note that this recurrent input is the filtered spike train $\mathbf{r}$, not the raw spikes $\mathbf{o}$. As a result, these new connections have slower dynamics than the connections presented in the first section. This motivates us to characterize connections as fast and slow depending on their underlying dynamics. The dynamics of the membrane potentials are now described by:

$$\dot{\mathbf{V}} = -\lambda_V\mathbf{V} + \mathbf{F}\mathbf{c} + \mathbf{W}^{\mathbf{s}}\mathbf{r} + \mathbf{W}^{\mathbf{f}}\mathbf{o} \tag{3}$$

where λ_V is the leak in the membrane potential, which is different from the leak in the decoder λ. It is clear from the previous construction that the slow connectivity $\mathbf{W^s} = \mathbf{F}(\mathbf{A} + \lambda\mathbf{I})\mathbf{F}^T$, is involved in generating the temporal dynamics of $\mathbf{x}$. Owing to the slow connections, the network is able to generate autonomously the temporal dynamics of the output and thus, only needs the command $\mathbf{c}$ as an external input. For example, if $\mathbf{A} = \mathbf{0}$ (i.e. the network implements a pure integrator), $\mathbf{W^s} = \lambda\mathbf{F}\mathbf{F}^T$ compensates for the leak in the decoder by generating a positive feedback term that prevents the activity form decaying. On the other hand, the fast connectivity matrix $\mathbf{W^f} = -\mathbf{F}\mathbf{F}^T$, trained with the unsupervised, voltage-based rule presented previously, plays the same role as in the autoencoder; It insures that the global output and the global coding error of the network are available locally to each neuron.

3 Teaching the network to implement a desired dynamical system

Our aim is to develop a supervised learning scheme where a network learns to generate a desired output with an error feedback as well as a local learning rule. The learning rule targets the slow recurrent connections responsible for the generation of the temporal dynamics in the output, as seen in the previous section. Instead of deriving directly the learning rule for the recurrent connections, we first derive a learning rule for the matrix $\mathbf{A}$ of the linear dynamical system using simple results from control theory, and then we translate the learning to the recurrent network.

3.1 learning a linear dynamical system online

Consider the linear dynamical system $\dot{\hat{\mathbf{x}}} = \mathbf{M}\hat{\mathbf{x}} + \mathbf{c}$ where $\mathbf{M}$ is a matrix. We derive an online learning rule for the coefficients of the matrix $\mathbf{M}$, such that the output $\hat{\mathbf{x}}$ becomes after learning equal to the desired output $\mathbf{x}$. The latter undergoes the dynamics $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{c}$. Therefore, we define $\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}}$ as the error vector between the actual and the desired output. This error is fed to the mistuned system in order to correct and "guide" its behavior (Fig 3A). Thus, the dynamics of the system with this feedback are $\dot{\hat{\mathbf{x}}} = \mathbf{M}\hat{\mathbf{x}} + \mathbf{c} + K(\mathbf{x} - \hat{\mathbf{x}})$, where K is a scalar implementing the gain of the loop. The previous equation can be rewritten in the following form:

$$\dot{\hat{\mathbf{x}}} = (\mathbf{M} - K\mathbf{I})\hat{\mathbf{x}} + \mathbf{c} + K\mathbf{x} \tag{4}$$

where $\mathbf{I}$ is the identity matrix. If we assume that the spectra of the signals are bounded, it is straightforward to show, via a Laplace transform, that $\hat{\mathbf{x}} \to \mathbf{x}$ when $K \to +\infty$. The larger the gain of the feedback, the smaller the error. Intuitively, if K is large, very small errors are immediately detected and therefore, corrected by the system. Nevertheless our aim is not to correct the dynamical system forever, but to teach it to generate the desired output itself without the error feedback. Thus, the matrix $\mathbf{M}$ needs to be modified over time. To derive the learning rule for the matrix $\mathbf{M}$, we operate a gradient descent on the loss function $L = \mathbf{e}^T\mathbf{e} = \|\mathbf{x} - \hat{\mathbf{x}}\|^2$ with respect to the components of the matrix. The component M_{ij} is updated proportionally to the gradient of L,

$$\delta M_{ij} \propto -\frac{\partial L}{\partial M_{ij}} \propto (\frac{\partial \hat{\mathbf{x}}}{\partial M_{ij}})^T\mathbf{e} \tag{5}$$

To evaluate the term $\partial\hat{\mathbf{x}}/\partial M_{ij}$, we solve the equation 4 for the simple case were inputs $\mathbf{c}$ are constant. If we assume that K is much larger than the eigenvalues of $\mathbf{M}$, the gradient $\partial\hat{\mathbf{x}}/\partial M_{ij}$ is approximated by $\mathbf{E}_{ij}\hat{\mathbf{x}}$, where $\mathbf{E}_{ij}$ is a matrix of zeros except for component ij which is one. This leads to the very simple learning rule $\delta M_{ij} \approx \hat{x}_j e_i$, which we can write in matrix form as:

$$\delta\mathbf{M} \propto \mathbf{e}\hat{\mathbf{x}}^T \tag{6}$$

The learning rule is simply the outer product of the output and the error. To derive the learning rule we assume constant or slowly varying input. In practice, however, learning can be achieved also using fast varying inputs (Fig 3).

3.2 Learning rule for the slow connections

In the previous section we derived a simple learning rule for the state matrix $\mathbf{M}$ of a linear dynamical system. We translate this learning scheme into the recurrent network described in section 2. To this end, we need to determine two things. First, we have to define the form of the error feedback in the

recurrent network case. Second, we need to adapt the learning rule of the matrix of the underlying dynamical system to the slow weights of the recurrent neural network.

In the previous learning scheme the error is fed into the dynamical system as an additional input. Since the input weight vector of a neuron $\mathbf{F}_i$ defines the direction that is relevant for its "action" space, the neuron should only receive the errors in that direction. Thus, the error vector is projected on the feedforward weights vector of a neuron before being fed to it. Accordingly, equation 3 becomes:

$$\dot{\mathbf{V}} = -\lambda_V \mathbf{V} + \mathbf{F}\mathbf{c} + \mathbf{W}^{\mathbf{s}}\mathbf{r} + \mathbf{W}^{\mathbf{f}}\mathbf{o} + K\mathbf{F}\mathbf{e} \tag{7}$$

In the autoencoder, the membrane potential of a neuron represents the error between the input and the output of the entire network along the neuron's feedforward weight. With the addition of the dynamic error feedback and the slow connections, the membrane potentials now represent the error between the actual and the desired network output trajectories.

To translate the learning rule of the dynamical system into a rule for the recurrent network, we assume that any modification of the recurrent weights directly reflects a modification in the underlying dynamical system. This is achieved if the updates $\delta\mathbf{W}^{\mathbf{s}}$ of the slow connectivity matrix are in the form of $\mathbf{F}(\delta\mathbf{M})\mathbf{F}^T$. This ensures that the network always implements a linear dynamical system and guarantees that the analysis is consistent. The learning rule of the slow connections $\mathbf{W}^{\mathbf{s}}$ is obtained by replacing $\delta\mathbf{M}$ by its expression according to equation 6 in $\mathbf{F}(\delta\mathbf{M})\mathbf{F}^T$:

$$\delta\mathbf{W}^{\mathbf{s}} \propto (\mathbf{F}\mathbf{e})(\mathbf{F}\hat{\mathbf{x}})^T \tag{8}$$

According to this learning rule, the weight update between two neurons, δW^s_{ij}, is proportional to the error feedback $\mathbf{F}_i\mathbf{e}$ received as a current by the postsynaptic neuron i and to $\mathbf{F}_j\hat{\mathbf{x}}$, the output of the network projected on the feedforward weight of the presynaptic neuron j. The latter quantity is available to the presynaptic neuron through its inward fast recurrent connections, as shown for the autoencoder in Fig 1D.

One might object that the previous learning rule is not biologically plausible because it involves currents present separately in the pre- and post-synaptic neurons. Indeed, the presynaptic term may not be available to the synapse. However, as shown in the supplementary information of [15], the filtered spike train r_j of the presynaptic neuron is approximately proportional to $\lfloor \mathbf{F}_j\hat{\mathbf{x}} \rfloor_+$, a rectified version of the presynaptic term in the previous learning rule. By replacing $\mathbf{F}_j\hat{\mathbf{x}}$ by r_j in the equation 8 we obtain the following biologically plausible learning rule:

$$\delta W^s_{ij} = E_i r_j \tag{9}$$

Where $E_i = \mathbf{F}_i\mathbf{e}$ is the total error current received by the postsynaptic neuron.

3.3 Learning the underlying dynamical system while maintaining balance

For the previous analysis to hold, the fast connectivity $\mathbf{W}^f$ should be learned simultaneously with the slow connections using the learning rule defined by equation 2. As shown in the first section, the learning of the fast connections establishes a detailed balance on the level of the neuron and guarantees that the output of the network is available to each neuron through the term $\mathbf{F}_j\hat{\mathbf{x}}$. The latter is the presynaptic term in the learning rule of equation 8. Despite not being involved in the dynamics per se, these fast connections are crucial in order to learn any temporal dynamics. In other words, learning a detailed balance is a pre-requirement to learn dynamics with local plasticity rules in a spiking network. The plasticity of the fast connections remediate very quickly any perturbation to the balance caused by the learning of the slow connections.

3.4 Simulation

As a toy example, we simulated a 20-neuron network learning a 2D-damped oscillator using a feedback gain $K = 100$. The network is initialized with weak fast connections and weak slow connections. The learning is driven by smoothed gaussian noise as the command $\mathbf{c}$. Note that in the initial state there are no fast recurrent connection and the output of the network does not depend linearly on the input because membrane potentials are too hyperpolarized (Fig 3B). The network's output is quickly linearized through the learning of the fast connections (equation 2) by enforcing a balance on the membrane potential (Fig 3B): initial membrane potentials exhibit large fluctuations

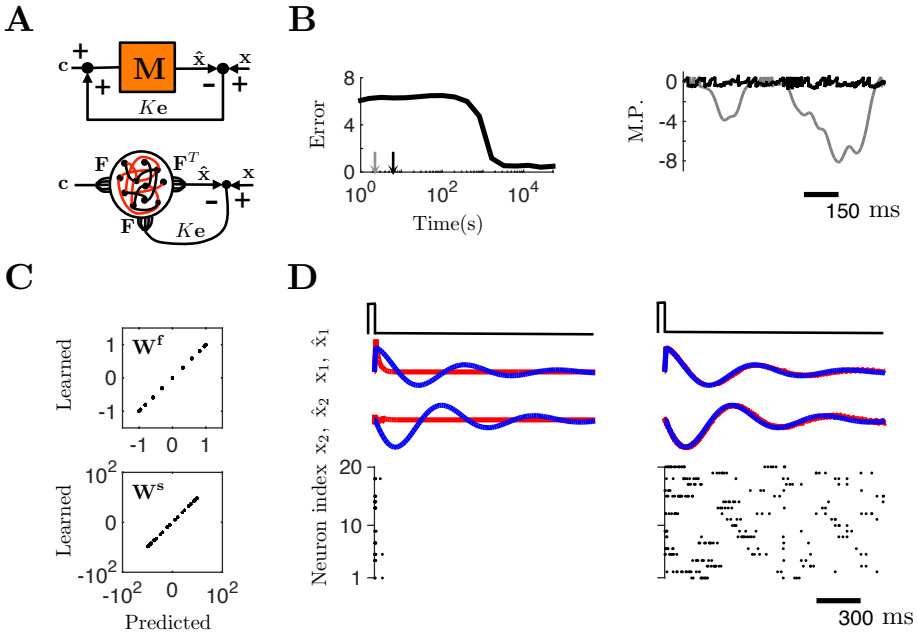

Figure 3: Learning temporal dynamics in a recurrent network. **A**, top panel: the linear dynamical system characterized by the state matrix $\mathbf{M}$ receives feedback signaling the difference between its actual output and a desired output. Bottom panel: a recurrent network displaying slow and fast connections is equivalent to the top architecture if the error feedback is fed into the network through the feedforward matrix $\mathbf{F}$. **B**: a 20 neuron network learns using equations 9 and 2. Left panel: the evolution of the error between the desired and the actual output during learning. The black and grey arrows represent instances where the time course of the membrane potential is shown in the next plot. Right panel: the time course of the membrane potential of one neuron at two different instances during learning. The gray line corresponds to the initial state while the black line is a few iterations after. **C**: scatter plots of the learned versus the predicted weights at the end of learning for fast (top panel) and slow (bottom panel) connections. **D**, top panels: the output of the network (red) and the desired output (blue), before (left) and after (right) learning. The black solid line on the top shows the impulse command that drives the network. Bottom panels: raster plots before and after learning. In the left raster plot there is no spiking activity after the first 50 ms.

that wane drastically after a few iterations (Fig 3B). On a slower time scale the slow connections learn to minimize the prediction error using the learning rule of equation 9. The error between the output of the network and the desired output decreases drastically (Fig 3B). To compute this error, different instances of the connectivity matrices were sampled during learning. The network was then re-simulated using the same instances while fixing K=0 to mesure the performance in the absence of feedback. At the end of learning, both slow and fast connections converge to their predicted values $\mathbf{W^s} = \mathbf{F}(\mathbf{A} + \lambda\mathbf{I})\mathbf{F}^T$ and $\mathbf{W^f} = -\mathbf{F}\mathbf{F}^T$ (Fig 3C). The presence of feedback is no longer required for the network to have the right dynamics (i.e. if we set $K = 0$ we still obtain the desired output, see Figs. 3D and 3B). The output of the network is very accurate (representing the state $\mathbf{x}$ with a precision of the order of the contribution of a single spike), parsimonious (no unnecessary spikes are emitted to represent the dynamical state at this level of accuracy) and the spike trains are asynchronous and irregular. Note that because the slow connections are weak at the initial state, spiking activity decays quickly once the command impulse is turned off, due to the absence of slow recurrent excitation (Fig 3D).

Simulation parameters Figure 1 : $\lambda = 0.05$, $\beta = 0.51$, learning rate: 0.01. Figure 3 : $\lambda = 50$, $\lambda_V = 1$, $\beta = 0.52$, $K = 100$, learning rate of the fast connections: 0.03, learning rate of the slow connections: 0.15.

4 Discussion

Using a top-down approach we derived a pair of spike-based and current-based plasticity rules that enable precise supervised learning in a recurrent network of LIF neurons. The essence of this approach is that every neuron is a precise computational unit that represents the network error in a subspace of dimension 1 in the output space. The precise and distributed nature of this code allows the derivation of local learning rules from global objectives.

To compute collectively, the neurons need to communicate to each other about their contributions to the output of the network. The fast connections are trained in an unsupervised fashion using a spike-based rule to optimize this communication. It establishes this efficient communication by enforcing a detailed balance between excitation and inhibition. The slow connections however are trained to minimize the error between the actual output of the network and a target dynamical system. They produce currents with long temporal correlations implementing the temporal dynamics of the underlying linear dynamical system. The plasticity rule for the slow connections is simply proportional to an error feedback injected as a current in the postsynaptic neuron, and to a quantity akin to the firing rate of the presynaptic neuron. To guide the behavior of the network during learning, the error feedback must be strong and specific. Such strength and specialization is in agreement with data on climbing fibers in the cerebellum [18–20], which are believed to bring information about errors during motor learning [21]. However, in this model, the specificity of the error signals are defined by a weight matrix through which the errors are fed to the neurons. Learning these weights is still under investigation. We believe that they could be learned using a covariance-based rule.

Our approach is substantially different from usual supervised learning paradigms in spiking networks since it does not target the spike times explicitly. However, observing spike times may be misleading since there are many combinations that can produce the same output [15, 16]. Thus, in this framework, variability in spiking is not a lack of precision, but is the consequence of the redundancy in the representation. Neurons having similar decoding weights may have their spike times interchanged while the global representation is conserved. What is important, is the cooperation between the neurons and the precise spike timing relative to the population. For example, using independent Poisson neurons with instantaneous firing rates identical to the predictive coding network drastically degrades the quality of the representaion [15].

Our approach is also different from liquid computing in the sense that the network is small, structured, and fires only when needed. In addition, in these studies the feedback error used in the learning rule has no clear physiological correlate, while here it is concretely injected as a current in the neurons. This current is used simultaneously to drive the learning rule and to guide the dynamics of the neuron in the short term. However, it is still unclear what the mechanisms are that could implement such a current dependent learning rule in biological neurons.

An obvious limitation of our framework is that it is currently restricted to linear dynamical systems. One possibility to overcome this limitation would be to introduce non-linearities in the decoder, which would translate into specific non-linearities and structures in the dendrites. A similar strategy has been employed recently to combine the approach of predictive coding and FORCE learning [7] using two compartment LIF neurons [22]. We are currently exploring less constraining forms of synaptic non-linearities, with the ultimate goal of being able to learn arbitrary dynamics in spiking networks using purely local plasticity rules.

Acknowledgments

This work was supported by ANR-10-LABX-0087 IEC, ANR-10-IDEX-0001-02 PSL, ERC grant FP7-PREDISPIKE and the James McDonnell Foundation Award - Human Cognition.

Refrences

[1] Kawato, M. (1999). Internal models for motor control and trajectory planning. *Current opinion in neurobiology*, 9(6), 718-727.

[2] Lackner, J. R., & Dizio, P. (1998). Gravitoinertial force background level affects adaptation to coriolis force perturbations of reaching movements. *Journal of neurophysiology*, 80(2), 546-553.

[3] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1988). Learning representations by back-propagating errors. *Cognitive* modeling, 5.

[4] Williams, R. J., & Zipser, D. (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural computation, 1*(2), 270-280.

[5] Jaeger, H. (2001). The echo state approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report, 148*, 34.

[6] Maass, W., Natschlger, T., & Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation, 14*(11), 2531-2560.

[7] Sussillo, D., & Abbott, L. F. (2009). Generating coherent patterns of activity from chaotic neural networks. *Neuron, 63*(4), 544-557.

[8] Legenstein, R., Naeger, C., & Maass, W. (2005). What can a neuron learn with spike-timing-dependent plasticity?. *Neural computation, 17*(11), 2337-2382.

[9] Pfister, J., Toyoizumi, T., Barber, D., & Gerstner, W. (2006). Optimal spike-timing-dependent plasticity for precise action potential firing in supervised learning. *Neural computation, 18*(6), 1318-1348.

[10] Ponulak, F., & Kasinski, A. (2010). Supervised learning in spiking neural networks with ReSuMe: sequence learning, classification, and spike shifting. *Neural Computation, 22*(2), 467-510.

[11] Memmesheimer, R. M., Rubin, R., lveczky, B. P., & Sompolinsky, H. (2014). Learning precisely timed spikes. *Neuron, 82*(4), 925-938.

[12] Gütig, R., & Sompolinsky, H. (2006). The tempotron: a neuron that learns spike timingbased decisions. *Nature neuroscience, 9*(3), 420-428.

[13] van Vreeswijk, C., & Sompolinsky, H. (1996). Chaos in neuronal networks with balanced excitatory and inhibitory activity. *Science, 274(5293)*, 1724-1726.

[14] Brunel, N. (2000). Dynamics of networks of randomly connected excitatory and inhibitory spiking neurons. *Journal of Physiology-Paris, 94*(5), 445-463.

[15] Boerlin, M., Machens, C. K., & Denève, S. (2013). Predictive coding of dynamical variables in balanced spiking networks. *PLoS computational biology, 9*(11), e1003258.

[16] Bourdoukan, R., Barrett, D., Machens, C. K & Denève, S. (2012). Learning optimal spike-based representations. In *Advances in neural information processing systems* (pp. 2285-2293).

[17] Vertechi, P., Brendel, W., & Machens, C. K. (2014). Unsupervised learning of an efficient short-term memory network. In *Advances in Neural Information Processing Systems* (pp. 3653-3661).

[18] Watanabe, M., & Kano, M. (2011). Climbing fiber synapse elimination in cerebellar Purkinje cells. *European Journal of Neuroscience, 34*(10), 1697-1710.

[19] Chen, C., Kano, M., Abeliovich, A., Chen, L., Bao, S., Kim, J. J., ... & Tonegawa, S. (1995). Impaired motor coordination correlates with persistent multiple climbing fiber innervation in PKC mutant mice. *Cell, 83*(7), 1233-1242.

[20] Eccles, J. C., Llinas, R., & Sasaki, K. (1966). The excitatory synaptic action of climbing fibres on the Purkinje cells of the cerebellum. *The Journal of Physiology, 182*(2), 268-296.

[21] Knudsen, E. I. (1994). Supervised Learning in the Brain. *The Journal of Neuroscience 14*(7), 39853997.

[22] Thalmeier, D., Uhlmann, M., Kappen, H.J., & Memmesheimer, R. (2015) Learning universal computations with spikes. *arXiv preprint arXiv:1505.07866*.

Universal Computations and Spike-Coding

The major limitation of the previous approach is that it only considers linear dynamical systems. Indeed, to implement interesting and complex tasks, the framework has to be extended to non-linear dynamical systems. A recent study (Thalmeier et al., 2015) succeeded in constructing such networks using the same spike-coding scheme used here. This success is due to the addition of non-linear dendrites to the network we presented. Next, we briefly present their approach.

If one assumes that instead of linear dynamics, $\mathbf{x}$ has general non-linear dynamics, $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{c}$ then, by accordingly replacing $\dot{\mathbf{x}}$ in the equation, 4.9 one obtains:

$$\begin{aligned}\dot{V}_n &= \mathbf{D}_n^T(\mathbf{f}(\mathbf{x}) + \mathbf{c} + \lambda\hat{\mathbf{x}} - \mathbf{D}\mathbf{o}) + \mu\lambda r_n - \mu o_n \\ &= \mathbf{D}_n^T\mathbf{f}(\mathbf{x}) + \mathbf{D}_n^T\mathbf{c} - \mathbf{D}_n^T\mathbf{D}^T\mathbf{o} + \lambda\mathbf{D}_n\hat{\mathbf{x}} + \mu\lambda r_n - \mu o_n\end{aligned}$$

By approximating $\mathbf{x} \approx \hat{\mathbf{x}} = \mathbf{D}\mathbf{r}$ and replacing in the previous equation, they obtain:

$$V_n = \mathbf{D}_n^T\mathbf{f}(\mathbf{D}\mathbf{r}) + \mathbf{D}_n^T\mathbf{c} - \mathbf{D}_n^T\mathbf{D}^T\mathbf{o} + \lambda\mathbf{D}_n\mathbf{D}\mathbf{r} + \mu\lambda r_n - \mu o_n \tag{4.13}$$

The authors then assume that the function $\mathbf{f}$ has the form:

$$\mathbf{f}(\mathbf{x}) = -\lambda_x\mathbf{x} + \mathbf{A}\tanh(\mathbf{x}) \tag{4.14}$$

where $\mathbf{A}$ is a matrix. Such form of non-linear functions is able to perform universal computation. In fact, the tanh function serves as a basis to approximate non-linear functions. By replacing $\mathbf{f}$ by its expression in equation 4.13 and rearranging the terms, they obtain:

$$V_n = \mathbf{D}_n^T\mathbf{A}\tanh(\mathbf{D}\mathbf{r}) - (\mathbf{D}_n^T\mathbf{D} + \mu\mathbf{I})\mathbf{o} + (a\mathbf{D}_n^T\mathbf{D} + \mu\lambda_s\mathbf{I}) + \mathbf{D}_n^T\mathbf{c} \tag{4.15}$$

where $a = \lambda - \lambda_x$. According to the previous equation, a neuron receives three types of recurrent inputs. The first term $\mathbf{D}_n^T\mathbf{A}\tanh(\mathbf{D}\mathbf{r})$ is akin to a non-linear dendrite that receives the postsynaptic currents $\mathbf{r}$ which are summed with weights $\mathbf{D}$. The sum is then subject to the non-linearity tanh. The dendritic output are conveyed to the soma with weights $\mathbf{D}_n^T\mathbf{A}$. The learning procedure concerns these dendro-somatic weights. To learn these weights the authors use the recursive least squares algorithm.

This network has proved to be capable of learning complex computations, such as periodic pattern generation, context dependent switching, non-linear control, etc. It is thus the first spiking recurrent network that is able to learn generic complex computations with this level of precision.

Chapter 5

Discussion

Throughout the three articles, we develop a theory of local learning in recurrent spiking networks. In the first two articles (Bourdoukan et al., 2012; Brendel et al., 2016), we study how a spiking recurrent network learns to efficiently represent its input using only local plasticity rules. The network has feedforward weights in addition to instantaneous recurrent connections. Both connections are plastic, so as to insure that the inputs are represented optimally. The learning in this auto-encoder solely depends on the input statistics and does not require the use of any supervising signal. In the third article (Bourdoukan and Denève, 2015), the network learns to implement a specific linear dynamical system. Its computational capabilities are enhanced by the addition of a slow recurrent connectivity and a supervising error signal.

5.1 Representing Information Within a Highly Cooperative Code

At the center of the learning scheme is the fact that global signals are represented locally and precisely in each neuron. Neurons are precise computational units that represent the signals in a subspace of dimension 1 in the global output space. This subspace, which can be called "feature", is defined by the encoding/decoding weight vector of the neuron (Boerlin et al., 2013). Because there are more neurons than input dimensions, decoding weights are generally non-orthogonal, resulting in an overlap between the features encoded by the different neurons. If two neurons have overlapping features and spike independently from one another (i.e, ignoring their mutual contribution to the output) they would contribute redundantly to represent the signal, which is a lack of efficiency. Thus, in the presence of overlapping features, an efficient representation requires a tight coordination of the spiking between neurons to avoid redundancy (Deneve and

Machens, 2016).

5.2 Fast Connections Enforce Efficiency by Learning to Balance Excitation and Inhibition

Fast recurrent connections enforce the efficiency of the representation by learning to balance the feedforward and recurrent inputs. At the end of learning, the recurrent input precisely cancels the feedforward charge accumulated by a neuron at the time scale of a single population's ISI. This results in a tightly balanced membrane potential over time. Because of this recurrent suppression, fast connections induce competition between the neurons that receive similar inputs. Such neurons encode for similar features. By suppressing each other's feedforward inputs, they compete to represent these features and thus avoid redundancy. Through learning, the gain in efficiency is such that coding accuracy strongly increases while firing rates are drastically reduced. Note that in the case of an over-complete representation, many neurons will represent similar features. Many spike sequences can encode a same signal at a comparable level of accuracy. Thus, the variability in spiking does not reflect a noisy representation but rather the degeneracy in spike patterns. The tight balance predicted in our model is in agreement with recent experimental evidence. (Okun and Lampl, 2008; Graupner and Reyes, 2013), showed that excitatory and inhibitory current received by nearby neurons are highly correlated over time. Inhibition appears to follow excitation with a lag a few milliseconds. Using a different method, such lag was also found in the auditory cortex. This delay opens a brief opportunity for the neurons to spike, resulting in a precise spike timing (Wehr and Zador, 2003; Marlin et al., 2015). Finally, strong correlations between excitation and inhibition are also observed during gamma oscillations in the hippocampus (Atallah and Scanziani, 2009).

Comparison to balanced networks Note that these networks, which were called spike-coding networks (Boerlin et al., 2013; Abbott et al., 2016; Thalmeier et al., 2015), are fundamentally different from earlier balanced recurrent networks (van Vreeswijk and Sompolinsky, 1996, 1998; Amit and Brunel, 1997; Brunel, 2000) for four main reasons. First, connectivity is not random and sparse but highly structured and dense. The weight between two neurons is equal to minus the product of their feedforward weight vectors. Indeed, the more the inputs received by two neurons are similar the more the suppression between them should be stronger. Second, the connection strength in these networks is much stronger and their strength is independent of connection density. In common balanced networks, the connection strength scales with $1/\sqrt{K}$, where K is the number of synapses

a neuron receives. Third, the balance in spike-coding networks is tighter. In earlier models, the balance exists only on average and excitation and inhibition are uncorrelated on a finer time scale. Fourth, the irregular spiking here is a signature of an efficient and highly coordinated spike-based code rather than a noisy independent Poisson code. Recent studies show that a tight balance can be achieved in both densely (Renart et al., 2010) and sparsely (Vogels et al., 2011) randomly connected networks. The latter uses an STDP kernel to drive the network into the detailed balance regime. However, none of these models could give such a straightforward and functional interpretation of tight balance in terms of coding.

Biophysical quantities have precise computational meaning The clear relationship between balance and coding is a consequence of the precise functional meaning that is borne by the biophysical quantities in this network. In fact, membrane potentials represent encoding errors, i.e. errors between the represented variable and the output of the network. Each neuron represents this multidimensional error along the direction specified by its encoding weight. Thus, low encoding errors are reflected by small and balanced membrane potentials. This means that recurrent activity, akin to the output, closely tracks the feedforward activity, akin to the input. Such a tight relationship between balance and coding allows the derivation of a local learning rule that enforces precise coding. Indeed, experiments have found that a tight balance between excitation and inhibition is a result of plasticity and is acquired through experience (Marlin et al., 2015; Dorrn et al., 2010; D'amour and Froemke, 2015; Froemke, 2015).

Local voltage based learning rules The learning rule for the fast connections is derived by minimizing the membrane potential fluctuations. This is equivalent to enforcing a balance, since high voltage fluctuations are the signature of imbalanced inputs. In a first attempt (Bourdoukan et al., 2012), we derived such a rule by minimizing in a greedy fashion the squared membrane potential at all times. This resulted in a rule proportional to the firing rate of the presynaptic neuron and to the voltage of the postsynaptic neuron. The network undergoing the learning rule indeed converges to the optimum. However the convergence is slow (Bourdoukan and Denève, 2015), Fig 1E. This slowness renders the learning of feedforward connections simultaneously, impossible. In fact, balance has to be restored quickly after the imbalance induced by the update of the feedforward weights. In the following paper (Brendel et al., 2016), we remedy this by minimizing the squared membrane potential only around spike times. The resulting rule depends on the postsynaptic membrane potential and on the synaptic weight. The voltage dependence of plasticity has been reported experimentally. Moreover, in an STDP-like experiment, the learning

rules reproduce the classical STDP windows. The shape of the widow (Hebbian or anti-Hebbian STDP) depends on whether the synapse is inhibitory of excitatory and on the target neuron. Indeed, the sub-threshold voltage dependence of plasticity has been established experimentally (Artola et al., 1990; Ngezahayo et al., 2000). In addition, other phenomenological voltage-based plasticity models were also able to reproduction STDP windows Clopath et al. (2010). Indeed, there is an intense debate on whether the spike timing dependence of plasticity is simply a consequence of the voltage dependence of plasticity (Lisman and Spruston, 2005).

We have previously discussed how fast connections enforced efficient encoding in the recurrent network by learning to balance excitation and inhibition (or alternatively feedforward and recurrent inputs). In fact, learning such a balance appears to be crucial in two different learning schemes: unsupervised learning of an auto-encoder (Brendel et al., 2016) and supervised learning of a linear dynamical system (Bourdoukan and Denève, 2015).

5.3 Learning an Autoencoder

Combining the previous rule with a Hebbian spike time-depenent rule for the encoding weights, allows the self-organization of an efficient auto-encoder. The learning rule implicitly pushes the feedforward weights to be aligned with the implicit decoder that is employed by the network. Indeed, for a fixed set of encoding weights, the plasticity of the fast connections sets the recurrent weights such that the neurons decode each other's activity using the same decoder $\mathbf{D}$. In a mathematical sense, this means that this decoder matrix is a factor of the recurrent connectivity matrix $\mathbf{W} = \mathbf{FD}$. More concretely, the implicit decoder enforced by fast plasticity is equal, for each neuron, to its spike-triggered average. Thus, the encoding weights are gradually pushed towards the spike-triggered average of each neuron, while the fast connections adapt quickly to this change. In the case of Gaussian inputs, the learning leads to optimal encoding/decoding weights. Such weights whiten the input and project it into orthogonal directions in the neural response space (Simoncelli and Olshausen, 2001).

Efficient coding, population coding and tuning curves After learning, the measured bell-shaped tuning curves reflect the input distribution. In the areas where the input is most probable, the tuning curves are dense and have low amplitude as opposed to the areas where inputs are less likely. (Ganguli and Simoncelli, 2014) shows that such a distribution of tuning curves results from maximizing the fisher information between the stimu-

lus and the responses. This work is founded on the classical population coding framework (Pouget et al., 2000) where the encoding model consists of fixed tuning curves that are dependent on the stimulus value. The neurons encode a stimulus value by firing spikes according to a Poisson process with a rate indicated by the tuning curve. However, in such a model, the neurons are assumed to fire independently from one another. This highly contrasts with the encoding scheme used by the tightly balanced auto-encoder. In the latter, neurons are highly competitive and coordinate their firing to represent the stimulus efficiently. The bell-shaped tuning curves are a result of the competition enforced by the learning of recurrent connections ((Brendel et al., 2016), Fig 2-3-S1). In this sense, the coding scheme is close to traditional efficient coding where neurons coordinate their responses to represent the stimulus by minimizing an objective (Olshausen and Field, 1996; Zylberberg et al., 2011). These models usually consider representations where neurons represent independent features. However, this does not account for the redundancy found in the brain. In contrast, here, similar features can be encoded by many neurons, resulting in the irregular spike trains. Thus this coding scheme is called efficient population coding (Deneve and Machens, 2016). A recent model succeeded in learning such an efficient sparse coding model using local learning rules derived form a single objective (Zylberberg et al., 2011). However, their interpretation of the optimization parameters in terms of biophysical quantities is less systematic and seems more of a heuristic approach. In our model, the functional meaning of biophysical quantities is better motivated and can thus be generalized to other types of computations.

5.4 Supervised learning

When the auto-encoder is endowed with an additional slower recurrent connectivity and a global error feedback, it is able to learn specific linear input-output transformation. The slow connections are trained to minimize the error between the actual and the desired output of the network. To guide the network's dynamics in the short term, this error is fed back and injected as a current in the neurons. The local plasticity rule for the slow weights is proportional to the firing rate of the presynaptic neuron and to the error current received by the postsynaptic neuron. The error feedback is not fed to the network with random weights, but along the encoding/decoding weight of each neuron. Indeed, a neuron should only be informed about errors that are relevant to its coding space. The strength and specificity of the error feedback are in agreement with data from the climbing fibers in the cerebellum. The learning of the slow and fast connections occurs simultaneously while the feedforward, the decoding and

the feedback connections are kept fix. The efficient coding scheme which is quickly enforced by the fast plasticity is crucial for the supervised learning of the slow connectivity. The learning of the fast connections precisely distributes the information concerning the output across neurons, rendering the use of local optimizing strategies possible for the slow weights.

Many studies that consider supervised learning in spiking neurons focus on reproducing precisely timed spike sequences(Memmesheimer et al., 2014; Ponulak and Kasiaski, 2010; Pfister et al., 2006; Gutig and Sompolinsky, 2006). Indeed, such a paradigm is irrelevant in our case where spike times are highly variable from trial to trial. This variability results from the degeneracy of the representation where many spike patterns lead to the same output. What is crucial is not the absolute timing of spikes but its coordination relatively to the population output. Instead of spike timing, the derivation of the learning rule relies on the underlying rate description of the network. In other words, it relies on its ability to relate discrete spikes to continuous representation (rates and dynamical variables). A recent study also makes the use of a rate description to define targets for the learning in spiking networks (DePasquale et al., 2016). This study succeeds in learning more complex tasks than we present here. However, it suffers from the use of non-local methods to train the network. For example, it uses an external auto-encoder of rate units to specify the target activity of the spiking network. In addition, the precision of these networks is considerably lower than the one achieved by the spike-coding scheme presented here. Another recent study (Thalmeier et al., 2015) succeeded to develop a network able to also learn highly complex functions using a similar coding strategy to the one presented here. The learning of non-linear computation is enabled by the use of •non-linear dendrites. The connections are trained using the FORCE Learning (Sussillo and Abbott, 2009) algorithm that was initially developed for networks of units with continuous activity. This is the first study that successfully applies the FORCE Learning method to spiking networks. This success is due to the natural rate description that underlies the dynamics of the network.

5.5 Open Questions

The efficient coding strategy employed by the network relies highly on the instantaneous nature of the fast connections. Any contribution to the output by a particular neuron is instantaneously transmitted to other neurons. Indeed, in biological networks, such instantaneous communication does not exist. In fact, the propagation of the action potential along the axon and the time course of PSP's result in delays of several milliseconds. A re-

cent study (Chalk et al., 2016) addressed this problem within the spike coding-scheme, and found that adding noise can remedy the synchronization induced by delays. From a different perspective, a possible solution would be to consider networks with sparser connectivity, thus limiting but indeed not suppressing the correlations between the membrane potentials. In this context, the spike coding and the tight balance hypothesis have to be revisited and adapted to these networks. For example the balance between excitation and inhibition in such network could be less tight but still be considerably tighter than in sparse randomly connected networks. On a learning level, this could imply that the rules should be derived by minimizing the membrane potential fluctuation on short horizon around spike rather sharply around spike times.

It is also interesting to extend the current auto-encoder in order to optimally learn inputs with non-Gaussian statistics. Indeed, the network proved to be efficient when trained on naturalistic signals. For instance, it learns to accurately represent speech signals, but the learning quality is less optimal than for Gaussian inputs.

In the current supervised learning scheme, the encoding, decoding and feedback weights are equal and are fixed throughout learning. This is a highly restrictive setup. It is thus crucial to understand how such an architecture can develop through learning. Moreover, the leaning that we consider is limited to linear dynamical systems. This considerably restricts the type of tasks that can be performed using this approach. Indeed, the actual framework should be extended to be able to perform non-linear functions. A possible solution is to adapt the learning scheme used here to spike coding networks with non-linear dendrites. These network were proved to be suitable for non-linear processing (Thalmeier et al., 2015).

Bibliography

Abbott, L., DePasquale, B., and Memmesheimer, R.-M. (2016). Building functional networks of spiking model neurons. *Nature neuroscience*, 19(3):350–355.

Abeles, M., Bergman, H., Margalit, E., and Vaadia, E. (1993). Spatiotemporal firing patterns in the frontal cortex of behaving monkeys. *Journal of neurophysiology*, 70(4):1629–1638.

Amit, D. J. and Brunel, N. (1997). Model of global spontaneous activity and local structured activity during delay periods in the cerebral cortex. *Cerebral cortex*, 7(3):237–252.

Anderson, J. S., Carandini, M., and Ferster, D. (2000). Orientation tuning of input conductance, excitation, and inhibition in cat primary visual cortex. *Journal of neurophysiology*, 84(2):909–926.

Artola, A., Bracher, S., and Singer, W. (1990). Different voltage-dependent thresholds for inducing long-term depression and long-term potentiation in slices of rat visual cortex. *Nature*, 347(6288):69–72.

Atallah, B. V. and Scanziani, M. (2009). Instantaneous modulation of gamma oscillation frequency by balancing excitation with inhibition. *Neuron*, 62(4):566–577.

Bair, W. and Koch, C. (1996). Temporal Precision of Spike Trains in Extrastriate Cortex of the Behaving Macaque Monkey. *Neural Computation*, 8(6):1185–1202.

Bell, C. C., Han, V. Z., Sugawara, Y., and Grant, K. (1997). Synaptic plasticity in a cerebellum-like structure depends on temporal order. *Nature*, 387(6630):278–281.

Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, 5(2):157–166.

Bi, G. Q. and Poo, M. M. (1998). Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience*, 18(24):10464–10472.

Bienenstock, E. L., Cooper, L. N., and Munro, P. W. (1982). Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex. *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience*, 2(1):32–48.

Blais, B. S., Intrator, N., Shouval, H., and Cooper, L. N. (1998). Receptive Field Formation in Natural Scene Environments: Comparison of Single-Cell Learning Rules. *Neural Computation*, 10(7):1797–1813.

Bliss, T. V. P. and Lomo, T. (1973). Long-lasting potentiation of synaptic transmission in the dentate area of the anaesthetized rabbit following stimulation of the perforant path. *The Journal of Physiology*, 232(2):331–356.

Boerlin, M. and Denève, S. (2011). Spike-based population coding and working memory. *PLoS Comput Biol*, 7(2):e1001080.

Boerlin, M., Machens, C. K., and Denève, S. (2013). Predictive coding of dynamical variables in balanced spiking networks. *PLoS Comput Biol*, 9(11):e1003258.

Borg-Graham, L., Monier, C., and Fregnac, Y. (1996). Voltage-clamp measurement of visually-evoked conductances with whole-cell patch recordings in primary visual cortex. *Journal of Physiology-Paris*, 90(3):185–188.

Bourdoukan, R., Barrett, D., Deneve, S., and Machens, C. K. (2012). Learning optimal spike-based representations. In *Advances in neural information processing systems*, pages 2285–2293.

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

Brendel, W., Bourdoukan, R., Vertechi, P., Christian, M., and Deneve, S. (2016). Reducing the dimensionality of data with neural networks. *Submitted*.

Britten, K. H., Shadlen, M. N., Newsome, W. T., and Movshon, J. A. (1992). The analysis of visual motion: a comparison of neuronal and psychophysical performance. *The Journal of Neuroscience*, 12(12):4745–4765.

Brunel, N. (2000). Dynamics of sparsely connected networks of excitatory and inhibitory spiking neurons. *Journal of computational neuroscience*, 8(3):183–208.

Brunel, N. and Nadal, J.-P. (1998). Mutual information, fisher information, and population coding. *Neural Computation*, 10(7):1731–1757.

Buracas, G. T., Zador, A. M., DeWeese, M. R., and Albright, T. D. (1998). Efficient Discrimination of Temporal Patterns by Motion-Sensitive Neurons in Primate Visual Cortex. *Neuron*, 20(5):959–969.

Cardin, J. A., Palmer, L. A., and Contreras, D. (2007). Stimulus feature selectivity in excitatory and inhibitory neurons in primary visual cortex. *The Journal of Neuroscience*, 27(39):10333–10344.

Celikel, T., Szostak, V. A., and Feldman, D. E. (2004). Modulation of spike timing by sensory deprivation during induction of cortical map plasticity. *Nature Neuroscience*, 7(5):534–541.

Chalk, M., Gutkin, B., and Denève, S. (2016). Neural oscillations as a signature of efficient coding in the presence of synaptic delays. *eLife*, 5.

Clopath, C., Büsing, L., Vasilaki, E., and Gerstner, W. (2010). Connectivity reflects coding: a model of voltage-based STDP with homeostasis. *Nature Neuroscience*, 13(3):344–352.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.

D'amour, J. A. and Froemke, R. C. (2015). Inhibitory and excitatory spike-timing-dependent plasticity in the auditory cortex. *Neuron*, 86(2):514–528.

Deadwyler, S. A. and Hampson, R. E. (1997). The significance of neural ensemble codes during behavior and cognition. *Annual Review of Neuroscience*, 20:217–244.

deCharms, R. C. (1998). Information coding in the cortex by independent or coordinated populations. *Proceedings of the National Academy of Sciences*, 95(26):15166–15168.

deCharms, R. C. and Merzenich, M. M. (1996). Primary cortical representation of sounds by the coordination of action-potential timing. *Nature*, 381(6583):610–613.

Deneve, S., Latham, P. E., and Pouget, A. (1999). Reading population codes: a neural implementation of ideal observers. *Nature Neuroscience*, 2(8):740–745.

Deneve, S. and Machens, C. K. (2016). Efficient codes and balanced networks. *Nature Neuroscience*, 19(3):375–382.

DePasquale, B., Churchland, M. M., and Abbott, L. F. (2016). Using Firing-Rate Dynamics to Train Recurrent Networks of Spiking Model Neurons. *arXiv:1601.07620 [q-bio]*. arXiv: 1601.07620.

Dorrn, A. L., Yuan, K., Barker, A. J., Schreiner, C. E., and Froemke, R. C. (2010). Developmental sensory experience balances cortical excitation and inhibition. *Nature*, 465(7300):932–936.

Engert, F., Tao, H. W., Zhang, L. I., and Poo, M.-m. (2002). Moving visual stimuli rapidly induce direction sensitivity of developing tectal neurons. *Nature*, 419(6906):470–475.

Feldman, D. E. and Brecht, M. (2005). Map Plasticity in Somatosensory Cortex. *Science*, 310(5749):810–815.

Fino, E., Deniau, J.-M., and Venance, L. (2008). Cell-specific spike-timing-dependent plasticity in gabaergic and cholinergic interneurons in corticostriatal rat brain slices. *The Journal of physiology*, 586(1):265–282.

Fino, E., Glowinski, J., and Venance, L. (2005). Bidirectional activity-dependent plasticity at corticostriatal synapses. *The Journal of neuroscience*, 25(49):11279–11287.

Freedman, D. J., Riesenhuber, M., Poggio, T., and Miller, E. K. (2006). Experience-dependent sharpening of visual shape selectivity in inferior temporal cortex. *Cerebral Cortex*, 16(11):1631–1644.

Froemke, R. C. (2015). Plasticity of Cortical Excitatory-Inhibitory Balance. *Annual Review of Neuroscience*, 38(1):195–219.

Froemke, R. C. and Dan, Y. (2002). Spike-timing-dependent synaptic modification induced by natural spike trains. *Nature*, 416(6879):433–438.

Froemke, R. C., Poo, M.-m., and Dan, Y. (2005). Spike-timing-dependent synaptic plasticity depends on dendritic location. *Nature*, 434(7030):221–225.

Ganguli, D. and Simoncelli, E. P. (2014). Efficient Sensory Encoding and Bayesian Inference with Heterogeneous Neural Populations. *Neural Computation*, 26(10):2103–2134.

Georgopoulos, A. P. (1990). Neural Coding of the Direction of Reaching and a Comparison with Saccadic Eye Movements. *Cold Spring Harbor Symposia on Quantitative Biology*, 55:849–859.

Georgopoulos, A. P., Schwartz, A. B., and Kettner, R. E. (1986). Neuronal population coding of movement direction. *Science*, 233(4771):1416–1419.

Gerstein, G. L. and Mandelbrot, B. (1964). Random walk models for the spike activity of a single neuron. *Biophysical journal*, 4(1 Pt 1):41.

Gerstner, W., Kempter, R., van Hemmen, J. L., and Wagner, H. (1996). A neuronal learning rule for sub-millisecond temporal coding. *Nature*, 383(6595):76–78.

Gollisch, T. and Meister, M. (2008). Rapid neural coding in the retina with relative spike latencies. *science*, 319(5866):1108–1111.

Graupner, M. and Reyes, A. D. (2013). Synaptic input correlations leading to membrane potential decorrelation of spontaneous activity in cortex. *The Journal of Neuroscience*, 33(38):15075–15085.

Gray, C. M. and McCormick, D. A. (1996). Chattering cells: superficial pyramidal neurons contributing to the generation of synchronous oscillations in the visual cortex. *Science*, 274(5284):109.

Gutig, R. (2016). Spiking neurons can discover predictive features by aggregate-label learning. *Science*, 351(6277):aab4113–aab4113.

Gutig, R. and Sompolinsky, H. (2006). The tempotron: a neuron that learns spike timing–based decisions. *Nature Neuroscience*, 9(3):420–428.

Habenschuss, S., Puhr, H., and Maass, W. (2013). Emergence of optimal decoding of population codes through stdp. *Neural computation*, 25(6):1371–1407.

Haider, B., Duque, A., Hasenstaub, A. R., and McCormick, D. A. (2006). Neocortical network activity in vivo is generated through a dynamic balance of excitation and inhibition. *The Journal of neuroscience*, 26(17):4535–4545.

Han, V. Z., Grant, K., and Bell, C. C. (2000). Reversible associative depression and nonassociative potentiation at a parallel fiber synapse. *Neuron*, 27(3):611–622.

Hebb, D. O. (1949). *The organization of behavior: A neuropsychological theory*. Psychology Press.

Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science (New York, N.Y.)*, 313(5786):504–507.

Jaeger, H. (2001). The "echo state" approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, 148:34.

Jazayeri, M. and Movshon, J. A. (2006). Optimal representation of sensory information by neural populations. *Nature neuroscience*, 9(5):690–696.

Kappel, D., Nessler, B., and Maass, W. (2014). Stdp installs in winner-take-all circuits an online approximation to hidden markov model learning. *PLoS Comput Biol*, 10(3):e1003511.

Kempter, R., Gerstner, W., and Hemmen, J. L. v. (2001). Intrinsic Stabilization of Output Rates by Spike-Based Hebbian Learning. *Neural Computation*, 13(12):2709–2741.

Kerlin, A. M., Andermann, M. L., Berezovskii, V. K., and Reid, R. C. (2010). Broadly tuned response properties of diverse inhibitory neuron subtypes in mouse visual cortex. *Neuron*, 67(5):858–871.

Knudsen, E. I., du Lac, S., and Esterly, S. D. (1987). Computational maps in the brain. *Annual Review of Neuroscience*, 10:41–65.

Kording, K. P., Tenenbaum, J. B., and Shadmehr, R. (2007). The dynamics of memory as a consequence of optimal adaptation to a changing body. *Nature neuroscience*, 10(6):779–786.

Lazar, A., Pipa, G., and Triesch, J. (2009). Sorn: a self-organizing recurrent neural network. *Frontiers in computational neuroscience*, 3:23.

Lisman, J. and Spruston, N. (2005). Postsynaptic depolarization requirements for LTP and LTD: a critique of spike timing-dependent plasticity. *Nature Neuroscience*, 8(7):839–841.

Lu, J.-t., Li, C.-y., Zhao, J.-P., Poo, M.-m., and Zhang, X.-h. (2007). Spike-Timing-Dependent Plasticity of Neocortical Excitatory Synapses on Inhibitory Interneurons Depends on Target Cell Type. *The Journal of Neuroscience*, 27(36):9711–9720.

Maass, W., Natschlager, T., and Markram, H. (2002). Real-Time Computing Without Stable States: A New Framework for Neural Computation Based on Perturbations. *Neural Computation*, 14(11):2531–2560.

Mainen, Z. F. and Sejnowski, T. J. (1995). Reliability of spike timing in neocortical neurons. *Science*, 268(5216):1503.

Markram, H., Lübke, J., Frotscher, M., and Sakmann, B. (1997). Regulation of synaptic efficacy by coincidence of postsynaptic aps and epsps. *Science*, 275(5297):213–215.

Marlin, B. J., Mitre, M., D'amour, J. A., Chao, M. V., and Froemke, R. C. (2015). Oxytocin enables maternal behaviour by balancing cortical inhibition. *Nature*, 520(7548):499–504.

Martens, J. (2010). Deep learning via hessian-free optimization. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 735–742.

Martens, J. and Sutskever, I. (2011). Learning recurrent neural networks with hessian-free optimization. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1033–1040.

McClurkin, J. W., Optican, L. M., Richmond, B. J., and Gawne, T. J. (1991). Concurrent processing and complexity of temporally encoded neuronal messages in visual perception. *Science (New York, N.Y.)*, 253(5020):675–677.

Mehta, M. R., Barnes, C. A., and McNaughton, B. L. (1997). Experience-dependent, asymmetric expansion of hippocampal place fields. *Proceedings of the National Academy of Sciences*, 94(16):8918–8921.

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

Minsky, M. and Papert, S. (1969). *Perceptrons.* MIT press.

Monier, C., Fournier, J., and Frégnac, Y. (2008). In vitro and in vivo measures of evoked excitatory and inhibitory conductance dynamics in sensory cortices. *Journal of neuroscience methods*, 169(2):323–365.

Ngezahayo, A., Schachner, M., and Artola, A. (2000). Synaptic activity modulates the induction of bidirectional synaptic changes in adult mouse hippocampus. *The Journal of Neuroscience*, 20(7):2451–2458.

Oja, E. (1982). Simplified neuron model as a principal component analyzer. *Journal of mathematical biology*, 15(3):267–273.

O'Keefe, J. and Recce, M. L. (1993). Phase relationship between hippocampal place units and the EEG theta rhythm. *Hippocampus*, 3(3):317–330.

Okun, M. and Lampl, I. (2008). Instantaneous correlation of excitation and inhibition during ongoing and sensory-evoked activities. *Nature neuroscience*, 11(5):535–537.

Olshausen, B. A. and Field, D. J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609.

Pfister, J.-P., Toyoizumi, T., Barber, D., and Gerstner, W. (2006). Optimal Spike-Timing-Dependent Plasticity for Precise Action Potential Firing in Supervised Learning. *Neural Computation*, 18(6):1318–1348.

Ponulak, F. and Kasiaski, A. (2010). Supervised learning in spiking neural networks with ReSuMe: sequence learning, classification, and spike shifting. *Neural Computation*, 22(2):467–510.

Pouget, A., Dayan, P., and Zemel, R. (2000). Information processing with population codes. *Nature Reviews Neuroscience*, 1(2):125–132.

Pouget, A., Zhang, K., Deneve, S., and Latham, P. E. (1998). Statistically efficient estimation using population coding. *Neural computation*, 10(2):373–401.

Rao, R. P. N. and Sejnowski, T. J. (2001). Predictive Learning of Temporal Sequences in Recurrent Neocortical Circuits. In Bock, G. R. and Goode, J. A., editors, *Complexity in Biological Information Processing*, pages 208–233. John Wiley & Sons, Ltd.

Renart, A., De La Rocha, J., Bartho, P., Hollender, L., Parga, N., Reyes, A., and Harris, K. D. (2010). The asynchronous state in cortical circuits. *science*, 327(5965):587–590.

Requarth, T. and Sawtell, N. B. (2011). Neural mechanisms for filtering self-generated sensory signals in cerebellum-like circuits. *Current opinion in neurobiology*, 21(4):602–608.

Richmond, B. J., Optican, L. M., Podell, M., and Spitzer, H. (1987). Temporal encoding of two-dimensional patterns by single units in primate inferior temporal cortex. I. Response characteristics. *Journal of Neurophysiology*, 57(1):132–146.

Richmond, B. J., Optican, L. M., and Spitzer, H. (1990). Temporal encoding of two-dimensional patterns by single units in primate primary visual cortex. I. Stimulus-response relations. *Journal of Neurophysiology*, 64(2):351–369.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

Schwartz, A. B. (1994). Direct cortical representation of drawing. *Science (New York, N.Y.)*, 265(5171):540–542.

Shadlen, M. N. and Newsome, W. T. (1994). Noise, neural codes and cortical organization. *Current opinion in neurobiology*, 4(4):569–579.

Shadlen, M. N. and Newsome, W. T. (1998). The variable discharge of cortical neurons: implications for connectivity, computation, and information coding. *The Journal of neuroscience*, 18(10):3870–3896.

Shouval, H. Z., Bear, M. F., and Cooper, L. N. (2002). A unified model of nmda receptor-dependent bidirectional synaptic plasticity. *Proceedings of the National Academy of Sciences*, 99(16):10831–10836.

Shu, Y., Hasenstaub, A., and McCormick, D. A. (2003). Turning on and off recurrent balanced cortical activity. *Nature*, 423(6937):288–293.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

Simoncelli, E. P. and Olshausen, B. A. (2001). Natural image statistics and neural representation. *Annual review of neuroscience*, 24(1):1193–1216.

Sjöström, P. J. and Häusser, M. (2006). A cooperative switch determines the sign of synaptic plasticity in distal dendrites of neocortical pyramidal neurons. *Neuron*, 51(2):227–238.

Sjostrom, P. J., Turrigiano, G. G., and Nelson, S. B. (2001). Rate, timing, and cooperativity jointly determine cortical synaptic plasticity. *Neuron*, 32(6):1149–1164.

Softky, W. R. and Koch, C. (1993). The highly irregular firing of cortical cells is inconsistent with temporal integration of random epsps. *The Journal of Neuroscience*, 13(1):334–350.

Song, S., Miller, K. D., and Abbott, L. F. (2000). Competitive Hebbian learning through spike-timing-dependent synaptic plasticity. *Nature Neuroscience*, 3(9):919–926.

Sussillo, D. and Abbott, L. F. (2009). Generating coherent patterns of activity from chaotic neural networks. *Neuron*, 63(4):544–557.

Thalmeier, D., Uhlmann, M., Kappen, H. J., and Memmesheimer, R.-M. (2015). Learning universal computations with spikes. *arXiv preprint arXiv:1505.07866*.

Theunissen, F. and Miller, J. P. (1995). Temporal encoding in nervous systems: a rigorous definition. *Journal of computational neuroscience*, 2(2):149–162.

Tolhurst, D. J., Movshon, J. A., and Dean, A. F. (1983). The statistical reliability of signals in single neurons in cat and monkey visual cortex. *Vision Research*, 23(8):775–785.

Toyoizumi, T., Pfister, J.-P., Aihara, K., and Gerstner, W. (2004). Spike-timing dependent plasticity and mutual information maximization for a spiking neuron model. In *Advances in neural information processing systems*, pages 1409–1416.

Toyoizumi, T., Pfister, J.-P., Aihara, K., and Gerstner, W. (2005). Generalized bienenstock–cooper–munro rule for spiking neurons that maximizes information transmission. *Proceedings of the National Academy of Sciences of the United States of America*, 102(14):5239–5244.

Van Rossum, M. C., Bi, G. Q., and Turrigiano, G. G. (2000). Stable hebbian learning from spike timing-dependent plasticity. *The Journal of Neuroscience*, 20(23):8812–8821.

van Vreeswijk, C. and Sompolinsky, H. (1996). Chaos in neuronal networks with balanced excitatory and inhibitory activity. *Science*, 274(5293):1724.

van Vreeswijk, C. and Sompolinsky, H. (1998). Chaotic balanced state in a model of cortical circuits. *Neural computation*, 10(6):1321–1371.

Vogels, T., Sprekeler, H., Zenke, F., Clopath, C., and Gerstner, W. (2011). Inhibitory plasticity balances excitation and inhibition in sensory pathways and memory networks. *Science*, 334(6062):1569–1573.

Wang, H.-X., Gerkin, R. C., Nauen, D. W., and Bi, G.-Q. (2005). Coactivation and timing-dependent integration of synaptic potentiation and depression. *Nature neuroscience*, 8(2):187–193.

Wang, X.-J. (2002). Probabilistic decision making by slow reverberation in cortical circuits. *Neuron*, 36(5):955–968.

Wehr, M. and Zador, A. M. (2003). Balanced inhibition underlies tuning and sharpens spike timing in auditory cortex. *Nature*, 426(6965):442–446.

Wittenberg, G. M. and Wang, S. S.-H. (2006). Malleability of Spike-Timing-Dependent Plasticity at the CA3–CA1 Synapse. *The Journal of Neuroscience*, 26(24):6610–6617.

Wu, G. K., Arbuckle, R., Liu, B.-h., Tao, H. W., and Zhang, L. I. (2008). Lateral sharpening of cortical frequency tuning by approximately balanced inhibition. *Neuron*, 58(1):132–143.

Xue, M., Atallah, B. V., and Scanziani, M. (2014). Equalizing excitation-inhibition ratios across visual cortical neurons. *Nature*, 511(7511):596–600.

Yang, S. N., Tang, Y. G., and Zucker, R. S. (1999). Selective induction of LTP and LTD by postsynaptic [Ca2+]i elevation. *Journal of Neurophysiology*, 81(2):781–787.

Zemel, R. S., Dayan, P., and Pouget, A. (1998). Probabilistic Interpretation of Population Codes. *Neural Computation*, 10(2):403–430.

Zhang, L. I., Tao, H. W., Holt, C. E., Harris, W. A., and Poo, M.-m. (1998). A critical window for cooperation and competition among developing retinotectal synapses. *Nature*, 395(6697):37–44.

Zohary, E., Shadlen, M. N., and Newsome, W. T. (1994). Correlated neuronal discharge rate and its implications for psychophysical performance. *Nature*, 370(6485):140–143.

Zylberberg, J., Murphy, J. T., and DeWeese, M. R. (2011). A sparse coding model with synaptically local plasticity and spiking neurons can account for the diverse shapes of v1 simple cell receptive fields. *PLoS Comput Biol*, 7(10):e1002250.