Biologically plausible learning and dynamics in neural networks

Dissertation
zur
Erlangung des Doktorgrades (Dr. rer. nat.)
der
Mathematisch-Naturwissenschaftlichen Fakultät
der
Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von

Paul Züge
aus
Bonn

Bonn 2025

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Rheinischen Friedrich-Wilhelms-Universität Bonn

Gutachter/Betreuer:	Prof. Dr. Raoul-Martin Memmesheimer
Gutachter:	Prof. Dr. Moritz Helias
Tag der Promotion:	16.06.2025
Erscheinungsjahr:	2025

Abstract

The plasticity and dynamics of biological neural networks enable remarkable computations despite physiological constraints. Here we study plasticity and dynamics that satisfy such constraints using analytically tractable linear rate networks and numerical simulations.

Biologically plausible models of reinforcement learning must be local and able to learn from sparse, delayed rewards. Weight (WP) and node perturbation (NP) achieve this by correlating fluctuations in synaptic strength or neuronal activity, respectively, with reward changes. Because the number of weights massively exceeds the number of neurons, NP was believed to be far superior to WP and more likely neurobiologically realized. We develop a clear, mathematically-grounded understanding of these versatile learning rules applied to linear rate networks learning a student-teacher task. Our analytical results show that WP can perform similarly to or even better than NP for many temporally extended and low-dimensional tasks, which we confirm in simulations of more complex networks and tasks. We further find qualitative differences in the weight and learning dynamics of WP and NP that might allow to distinguish them experimentally. The generated insights allow us to formulate modified learning rules that in certain situations combine the advantages of WP and NP. Together, our findings indicate WP as competitive or even preferable to NP for many relevant biological and machine learning tasks, suggesting it as a useful benchmark and plausible candidate for learning in the brain.

Biologically plausible models of neural computation must reflect experimentally observed network characteristics. One such characteristic is that principal neurons in sensory cortices encode continuous variables with overlapping responses, featuring predominant excitation between neurons with strongly overlapping responses. The reasons underlying such connectivity are still unclear, and there are even known disadvantages to it. To address this knowledge gap, we develop a novel cooperative coding scheme that relies on like-to-like excitation to implement a desired response. Neurons cooperatively share their computations and access to feedforward input with similarly-tuned neurons that also need them. This allows to exchange many feedforward and less specific recurrent connections for few specific recurrent ones, thereby minimizing the total number of synapses. By comparing cooperatively coding and feedforward networks achieving the same network response, we find that synaptic savings come at the cost of increased network response times. This trade-off improves in magnitude and scaling when introducing delayed, balancing inhibition or spike frequency adaptation, or when encoding higher-dimensional stimuli. Our results suggest the number of synapses as an important constraint that can explain observed connectivity patterns in a novel cooperative coding scheme, possibly enabled by balancing inhibition.

Acknowledgements

I want to express my sincere gratitude to my supervisor Raoul-Martin Memmesheimer: for the interesting research opportunities, his availability and support, the many discussions and shared ideas, the conference visits, his sense of humor and, last but not least, his financial support. Further, I want to thank Moritz Helias for agreeing to be my second reviewer, and Wolfram Barfuss and Johann Kroha for rounding out the doctoral committee. Johann Kroha I also thank again for having supervised my Master's thesis.

I thank Christian Klos for the fruitful collaboration on our node- and weight perturbation project, for his generous proofreading of this thesis and for fun runs. I am also grateful for Natalie Schieferstein's ongoing collaboration on our cooperative coding project (in revision) and for her inspiring scientific rigor. The early supervision and guidance of Aditya Gilra is much appreciated.

Heartfelt thanks go to my group: To my longtime office mate Sven Goedeke, who always fueled my excitement over new ideas with relevant literature and exciting discussions, as well as with his own enthusiasm. To Felipe Kalle Kossio for always spicing things up and being 'the nicest person in the office'; also for important discussions and getting me to try apnea diving. To Simon Altrogge for being consistently helpful and taking initiative for the group, and Clemens Engelhardt, who deeply enriched our coffee culture. To Paul Manz, for his serenity and 7-Wonders initiatives, and Wilhelm Braun, for after-work-gatherings. To Carlo Wenig, for his calm helpfulness, and Jan Ahlborn, for letting me supervise his Bachelor's thesis. My warm thanks extend to all other former group members.

Throughout the years, Simone Christian helped me more than once with assistance that was simultaneously fast, cheap, and excellent. I also owe great thanks to Walter Witke for supporting me, as well as our group in general.

My friend Dr. Simon Essink has been an inspiration, and a great help through his proofreading, literature suggestions and many discussions. I thank Max Dax and Paul Klingenmayer, first of all for being very great friends, and also for fun home office at their places. This phase in Bonn has been particularly special, thanks to my wonderful friends Thorsten, Dela, Niki, Felix, Mathias, Fredi, Eva, Elisabeth, Aysel, Yannick, Konstantin, and Dinah. To Anja, Michael, Jörg, Timo, Nadine, and Dinah: thank you for challenging and inspiring me, and helping me grow into the person I am today.

I am deeply grateful for my loving and caring parents, who always gave me both freedom and safety to do what I want. And I am grateful to have, and to have grown up with, such a great brother as Henning, who inspires me in many ways. My wider family is also very dear to me, und ich möchte an dieser Stelle Oma Lorchen danken, deren warmherzige Art sich durch die Generationen zieht, und die mich bei meiner Arbeit regelmäßig an meine Prioritäten erinnert.

I am more than grateful to have you, Dinah, in my life. Thank you for accepting me as I am, yet encouraging me to become what I can.

List of Publications

[1] **P. Züge**, C. Klos and R.-M. Memmesheimer
Weight versus Node Perturbation Learning in Temporally Extended Tasks: Weight Perturbation Often Performs Similarly or Better
Physical Review X **13** (2023) 021006

[2] **P. Züge** and R.-M. Memmesheimer
Cooperative Coding of Continuous Variables in Networks with Sparsity Constraint
bioRxiv (2024):2024.05.13.593810v2, preprint

Contents

CHAPTER 1

Introduction

The brain is the physical substrate that allows us humans and other animals to perceive the world, make sense of it and act in it. How it does so is still a big mystery. There is however a consensus that neurons and their interactions with each other are central to its computational capabilities [3, 4]. In this view, neurons perform elementary computations by integrating input from other neurons and translating it into output, which becomes the input to yet other neurons. By chaining these elementary computations and through emergent network effects, the neural network can implement much more complex and powerful computations than its constituents [5]. Network function is thereby largely determined by the pattern and strengths of the synapses that connect neurons with each other, and plasticity in these is widely considered as the main contributor to learning [6, 7].

The result of network-level computations involves the coordinated activity of many neurons, such as for example motor neurons in the execution of movements. It can thereby be helpful to think of the neurons as independent agents that work together towards a network-level goal that does not exist on the neuron-level: In contrast to artificial neural networks, which are simulated on computers that have unrestricted access to their full dynamic states, biological neurons can only access locally available information. Each neuron has to use this limited, 'private' information to produce and adapt its output. How does it contribute to the full network dynamics? And how can it change to improve the network? This question is known as the credit assignment problem [8], which is complicated by the mentioned locality constraint. Further constraints that shaped the evolution of biological neural networks include the availability of space and energy [9–11].

This thesis examines how local, biologically plausible computations can jointly achieve global objectives: On the level of plasticity, we study two stochastic reinforcement learning (RL) rules, Weight perturbation (WP) [12, 13] and node perturbation (NP) [14, 15], that leverage synapse-local information to maximize the reward obtained by the network (publication 1 [1], Chapter 3). The learning rules correlate random fluctuations in synaptic weights or neuronal activity, respectively, – which are another constraining and seemingly detrimental factor – with reward prediction errors to guide synaptic plasticity. On the level of activity, we develop a novel cooperative coding scheme, in which neurons utilize local recurrent connectivity as observed in cortex to dynamically generate a desired network response (publication 2 [2], Chapter 4). The coding scheme thereby requires a minimal number of synapses, suggesting that synapse number and limited space are important constraints shaping neural networks.

Fluctuation-driven reinforcement learning In addition to activity-dependent plasticity, synapses undergo apparently random, activity-independent plasticity that is similar in magnitude [16–19]. Long-term synaptic changes can be consolidated or reverted at a later point in time [20–22]. This consolidation is influenced by neuromodulators such as dopamine [22–24], which encodes reward- and general prediction errors [25]. It can be accessible to all synapses in an area through diffusive volume transmission [26], giving them access to a 'global' reward signal [27]. WP and NP are two basic RL rules that make use of random fluctuations in neural networks, requiring only a global reward signal in addition to principally locally-available information [28]. They work by correlating perturbations of the synaptic strengths (weights) or summed synaptic inputs (nodes) with changes in the obtained reward, increasing the probability to produce activity that lead to higher rewards [29] (App. C). Their updates thereby follow noisy, unbiased estimates of the weight gradient [13, 14].

Because there are typically many more weights than nodes, the perturbation space that WP uses to estimate the weight gradient is (for time-independent tasks) much larger than that of NP. NP therefore has the reputation of being far superior to WP [14] and is more often considered as actually implemented in the brain [28, 30, 31]. One example is birdsong learning, where the synapses from a conducting area (HVC) that provides highly precise and sparse, clock-like inputs [32] to an output area (RA) receive 'perturbing' inputs from an experimenter area (LMAN) that might function as node perturbations [30, 33]. NP is also more often used as a benchmark [34–36].

The theoretical work that showed NP's superiority studied linearly mapping random inputs to outputs as given by a teacher network [14]. Solving this task requires matching every weight of the teacher network, because each weight has a statistically independent effect on its output neuron. In contrast, neurobiological tasks and neuronal activity are typically (relatively) low-dimensional [37–39], with many weight configurations providing solutions. This suggests that WP can learn such tasks more easily, as performance depends effectively on fewer (combinations of) weights. Also, tasks extend in time. This increases the perturbation dimension of NP but not WP In order to be able to learn arbitrary output sequences, NP has to perturb the output nodes independently at different time points. In contrast, WP can apply weight perturbations that are constant throughout a trial, capitalizing on the fact that neuronal networks can generate neuronal dynamics with static weights.

Inspired by these observations, we study WP and NP in settings that feature temporally-extended and correlated neuronal activity. Through analyzing them on analytically tractable tasks we generate a mathematically grounded understanding of the dependence of their learning characteristics on different task parameters. Our findings of comparable or superior performance of WP, also for more complex networks and tasks, suggest reconsidering WP as actually implemented in the brain, as well as using it as a relevant benchmark for other biologically plausible RL rules. As WP and NP are massively parallelizable and therefore potentially relevant for machine learning applications [40], our results might motivate their use for long tasks where traditional gradient-computation is either too costly or difficult due to long time horizons. The developed insights and learning rule variants might enable further improving them.

Cooperative coding of continuous variables Cortical principal neurons involved in the processing of continuous sensory variables encode these with overlapping response profiles [41, 42]. Highly similarly responding neurons are thereby more likely to excite each other [43–45], and do so more strongly [44]. Also their functional connectivity is excitatory [46, 47]. The reason for such like-to-like excitation is unclear: models optimized to encode variables with few spikes feature like-to-like inhibition

instead [48, 49]. Further, recurrent excitation can amplify noise [50] and increase response times [51, 52].

This poses the question: What is the reason underlying the observed like-to-like excitation? Theoretical neuroscience can provide possible explanations to such normative questions by showing that the observed features emerge when optimizing models under some constraint or objective. Optimal usage of limited space can for example predict the ratios of the volumes taken up by neuronal wiring to neuronal cell bodies [53], of excitatory to inhibitory neurons [11], or the distribution of axon diameters [54].

In line with this tradition, we show that excitation between similarly-tuned neurons can be a sign of an architecture that minimizes the number of required synapses. Concretely, we develop a cooperative coding scheme in which neurons use these recurrent connections to dynamically generate the network response (publication 2 [2], Chapter 4). The recurrent interactions thereby allow the cooperative sharing of already performed computations with similarly-tuned neurons that also need them. They further distribute access to feedforward input across the network. We show that this recurrent sharing allows to save many synapses, particularly compared to a pure feedforward implementation. As recurrent neural networks are formally equivalent to deep feedforward networks that are 'unrolled in time' [55], the addition of recurrent connections allows further nonlinear computations in networks with nonlinear neuronal activations. We do not consider this effect, but use linear rate neurons to show how different sets of inputs can generate a desired response. The computational benefits of recurrent connections are then in addition to the synaptic savings, and might even lead to further savings [56].

The synaptic savings in our cooperatively coding networks come at the cost of slower response formation. The network dynamics can be sped up when excitation is balanced by delayed inhibitory currents. This allows an increase in excitation, which in the stationary state cancels with the additional inhibition. During the response formation, however, excitation rises before the lagged inhibition, creating a 'window of opportunity' during which strong, yet unbalanced excitation can quickly propagate activity through the network. This ability of balanced networks to amplify their input without significant slowing of their dynamics relies on an effect called balanced amplification [51]: a surplus of excitation strongly and transiently drives both excitatory and inhibitory activity, which then decay quickly due to dominating inhibition. Cortex operates in such a regime of strong recurrence, in which supercritical recurrent excitation would, on its own, quickly cause 'exploding activity', but is stabilized by inhibition [57–59]. This E/I balance can also explain spike train characteristics: in a dynamically maintained regime called the balanced state excitation and inhibition largely cancel, and the remaining fluctuations cause asynchronous, irregular neuronal firing [60–62]. Experiments find that excitation and inhibition are indeed correlated, and that inhibition tracks excitation with a short delay [63–65] as utilized in our balanced cooperative coding networks. Delayed, balancing inhibitory currents can also arise as spike-frequency adaptation currents, a feature often observed in excitatory principal neurons [7, 66, 67].

Thesis structure Chapter 2 provides foundational information about the neurobiological and methodological background relevant to the thesis. It covers biological neural networks (Sec. 2.1), coding in recurrent networks (Sec. 2.2), neural networks as dynamical systems (Sec. 2.3), fluctuation-driven reinforcement learning (Sec. 2.4) and some mathematical tools (Sec. 2.5). Chapter 3 contains a brief summary of publication 1 [1], which is attached in App. A. Chapter 4 briefly summarizes publication 2 [2], which is attached in App. D. Both publications are discussed in Chapter 5. App. C contains an additional analysis of the relationship between WP and the REINFORCE framework [29].

CHAPTER 2

Foundations

2.1 Biological neural networks

This section gives a brief introduction to biological neural networks, emphasizing the aspects relevant to this thesis.

2.1.1 The nervous system

The human nervous system consists of two parts: the central nervous system (CNS) and the peripheral nervous system (PNS) [3, 68]. The CNS is encased in bone and consists of the brain and spinal cord [68]. All other parts of the nervous system make up the PNS. It connects the CNS with the body, via nerves (axon bundles) that enter and leave the spinal cord. Additionally, there are the cranial nerves, innervating mostly the head, of which some belong to the CNS and others to the PNS [68].

This work focuses on the brain, more specifically on the cerebral cortex. This layered structure at the outside surface of the brain is especially large in humans and critical for higher cognitive functions [68].

The brain contains two main cell classes: neurons and glial cells [4]. In the human brain, it was believed that around 75 to 80 % of cells are glial cells [3, 4], but improved counting methods estimate a one-to-one ratio [69]. Neurons are the highly specialized, electrically excitable cells that enable neuronal computation through their generation and propagation of electrical signals. Glial cells do not directly take part in electrical signaling, but support neuronal functioning in various ways, such as nourishing neurons, regulating extracellular concentrations of ions and neurotransmitters, or myelinating axons [3, 4].

2.1.2 Neurons

Neurons can process chemical and electrical signals and transmit information over long distances [6]. Their importance further derives from the fact that they are, as part of the nervous system, involved in sensing and control muscles and glands [68]. They consist of a cell body, also called soma, tree-like structures called dendrites, and an axon [4, 6]. The dendrites integrate and relay inputs to the soma, which generates output in the form of short action potentials or spikes if its electric potential exceeds a firing threshold. The action potentials then travel along the axon and are transmitted to other neurons via connecting synapses.

Electrical properties A neuron processes electrical signals by manipulating its electrical potential relative to the surrounding extracellular fluid, in response to chemical and other inputs [6]. This potential, termed membrane potential, stems from the different concentrations of various ions on the inside and outside surface of its cell membrane. It is typically negative, meaning that there are excess negative charges on the inside. As like charges repel, these gather on the inside surface of the membrane. Electrostatic forces attract an equal amount of positive charges to the outside surface, establishing electrical neutrality on a broader scale. The insulating cell membrane separates the charges and acts as a capacitor. However, ions can traverse it through a range of ion channels, thereby increasing (depolarizing) or further decreasing (hyperpolarizing) the membrane potential [6]. For each ion there is an equilibrium potential at which the in- and outward currents due to electrostatic and diffusive forces cancel. Ionic currents with equilibrium potentials above a neuron's firing threshold, such as Na^+ and Ca^{2+}, increase the likelihood of firing and are termed excitatory. Currents with subthreshold equilibrium potentials, such as K^+ and Cl^-, hinder firing and are called inhibitory [6]. Ion channels are typically permeable to multiple ion types. The sum of the ionic currents then switches sign at the so-called reversal potential, which lies in between the involved equilibrium potentials.

The time-independent component of a neuron's conductivities for different ion types can be captured in its leak conductivity and related resting potential. The resting potential is the potential that the membrane potential relaxes back to in the absence of input, with a speed determined by the leak conductance [6]. Conductivities, however, vary with time, as channels can open or close depending on the membrane potential, on the presence of neurotransmitters or on intracellular concentrations of specific ions or messenger molecules [6]. This allows neurons to respond to input signals in a complex way, in particular by translating synaptic events into electrical signals and by generating output spikes.

A spike, also called an action potential, is a short, stereotyped membrane potential deflection that lasts about 1 ms [6]. It is triggered by the dynamics of voltage-gated Na^+ and K^+ channels and involves a fast voltage increase of about 100 mV, followed by a re- and afterhyperpolarization. After a spike, there is a short period of absolute refractoriness in which it is impossible to elicit another spike, followed by a relative refractory period in which spiking is possible but hampered [6]. A spike is triggered at the axon initial segment when the membrane potential of a neuron surpasses a critical threshold for spiking [4]. It then propagates along the axon, where it is actively reinforced [4]. Subthreshold fluctuations in the membrane potential, on the other hand, are not reinforced but strongly attenuate along the axon [6]. This, together with the highly stereotyped form of the action potential, means that the only aspect of the activity of a neuron that determines its axon-mediated influence on other neurons is its spike times.

Axon The axon is a long, cable-like process emerging from the soma that transmits action potentials to other neurons. The spike propagation along the axon can be likened to a wildfire, using the same dynamics of voltage-gated Na^+ and K^+ channels as spike generation. Similar to a fire front, propagating spikes cannot reverse direction, as the Na^+ channels behind them are still inactivated [6]. Axons have typical lengths on the order of millimeters [3] to centimeters, much longer than the typical reach of dendrites [6]. Axons connecting distant brain areas can be even longer.

Synapses A synapse is a connection between two neurons that allows the presynaptic neuron to influence its postsynaptic partner. The effect depends on the type of synapse and on its strength. In this way, the presence or absence of synapses, i.e., the connectivity, determines which neurons a neuron can directly influence, and the heterogeneity of synaptic strengths allows it to affect its synaptic targets

differentially. Synaptic connectivity and connection strengths thereby largely determine network function. The plasticity of synaptic connectivity and synaptic strengths is widely recognized as the main contributor to learning [6].

Synaptic transmission works as follows: The presynaptic (axonal) terminal contains neurotransmitters packaged into vesicles [4]. A presynaptic spike arriving at the synapse prompts stochastic transmitters release via the fusion of docked vesicles with the cell membrane. The released neurotransmitters diffuse across the synaptic cleft, which is the small space of extracellular medium that separates the two neurons at the point of contact [4]. At the postsynaptic side, the neurotransmitters activate specialized receptors. These in turn cause the opening of ion channels, which translates the chemical signal into a postsynaptic potential (change) [4].

The postsynaptic potential can be either excitatory or inhibitory, with the postsynaptic receptor determining the synaptic effect. Dale's principle states that a neuron releases the same neurotransmitter or mix of neurotransmitters at all terminal sites [68], which holds with some exceptions for the whole brain [4]. Although any neurotransmitter can activate multiple receptor subtypes that have different postsynaptic effects [68], in practice most neurons can be classified as either excitatory or inhibitory [6]. The main excitatory neurotransmitter is glutamate [6, 68]. Prominent glutamate receptors are the fast AMPA receptors and the slower NMDA receptors [68]. For the main inhibitory neurotransmitter, γ-aminobutyric acid (GABA), there is also a receptor with a fast response ($GABA_A$) and another with a slow response ($GABA_B$) [6].

While most synapses are chemical synapses, as described above, some are electrical [4]. Electrical synapses directly connect the cytoplasm of the involved neurons, allowing a near instantaneous coupling of membrane potentials and transmission of subthreshold fluctuations [4].

Dendrites Depending on neuron type, a neuron can have one or more dendrites. These become thinner with distance from the soma, as the dendrite splits up into more and more branches. The branches often have small protrusions, called dendritic spines, that support a synapse by housing neurotransmitter receptors [70]. Their primary function is thought to be providing a microcompartment that segregates postsynaptic chemical responses, which are important to guide synaptic plasticity [70]. While excitatory synapses typically connect with dendritic spines, inhibitory synapses can target the dendrite directly, or also the soma or axon [4].

Synaptic transmission creates a peak in the local ion concentrations and electric potential. The ions follow the voltage gradient, thereby broadening the peak and decreasing the voltage gradient. The result is a diffusion-like current that gets attenuated with distance so that synapses closer to the soma have a stronger, more immediate and temporally defined effect [6]. However, there are compensatory mechanisms that can ensure that distal synapses still contribute measurably to neuronal firing [4, 71].

Although dendrites can perform complex computations on their own [71], they are typically modeled as linear integrators of their synaptic currents [6, 7]. This simplification captures the fact that dendrites do integrate their inputs spatially and temporally.

2.1.3 Neuron types

Neurons organize hierarchically into a large number of classes, subclasses, and cell types with shared morphological, electrophysiological, connectional and transcriptomic features. Two main classes are glutamatergic excitatory and GABAergic inhibitory neurons [6, 72]. Excitatory and inhibitory neurons appear in a ratio of approximately four to one [4, 11]. Although lower in numbers, there is a greater

diversity in inhibitory neurons [72]. Neurons can also be divided into principal/projection neurons and interneurons [4]. While interneurons only innervate neurons within their area, principal neurons also project to other areas and thus represent the output of their regions [4]. In the cerebral cortex, projection neurons are typically excitatory and interneurons typically inhibitory [4]. The major class of neocortical projection neurons are pyramidal cells (PCs), named after the pyramidal shape of their somata. Pyramidal cells and their dendritic differentiation are highly conserved across evolutionary distant classes such as mammals, birds, reptiles and fish, speaking for their adaptive value [68]. Making them even more interesting, they are found in evolutionary newer brain regions linked to higher cognitive functions.

2.1.4 Networks of neurons

Neurons can be seen as the basic computational units of the brain. However, they do not work in isolation but together in networks. As an example, the cortical column is often referred to as the smallest functional unit of the cortex [68].

The brain segregates into many specialized areas with sometimes strongly differing architectures, connectivity patterns and highly specialized neurons. The different areas of the neocortex, however, have a rather similar intra-areal structure but differ by their inter-areal inputs and projections. Just as neurons form local networks, these areas don't work in isolation, but many areas coordinate to give rise to a specific function, and one area often contributes to multiple functions [73]. Bottom-up input from one area to another, hierarchically higher area is called feedforward input, while top-down input is called feedback input; local input from within the same area is called recurrent input. There might not be one unifying principle of brain functioning; rather, different areas employ different computational and dynamical strategies. Similarly, known plasticity rules might apply to specific connection types in some brain areas but not others. Finally, dynamics and plasticity depend on brain states and context. This makes it important to refer to the specific area that a given model describes in a given context.

Despite the differences between brain areas, there are some general patterns. Somata and long-range axons segregate into regions with a high density of cell bodies and mainly unmyelinated axons, called gray matter, and regions containing nerve tracts of myelinated axons but few neurons, called white matter [3].

The cerebral cortex is an especially interesting structure as the seat of uniquely human reasoning and cognition [68]. It is common to all vertebrate animals and has conserved features that point to its evolutionary value: a layered organization with at least one layer containing pyramidal cells with characteristically differentiated dendrites [68]. It can be divided into sensory, motor and association areas [68]. The primary visual cortex (V1) as an example of a sensory area receives visual information from the retina via the thalamus [68]. In humans, association areas not directly related to sensory processing or motor control make up a large part of the cortex [68].

2.2 Neural coding

Neural coding addresses the question of how neuronal activity encodes information, and especially how it relates to stimuli [6, 74]: what information about a stimulus is encoded in what features of the activity of a neural network? When the network activity contains sufficient information, one can say that it encodes the stimulus properties, and that these can in turn be decoded from it.

One distinguishes different types of neural codes, depending on the presence of correlations between spikes, between neurons and on temporal precision. Codes in which correlations between the timings of multiple spikes of a neuron carry a significant amount of information — for example if information is encoded in interspike intervals — are called correlation codes, whereas in an independent-spike code such correlations are not informative [6]. On a population level, correlations between spike trains of different neurons may carry information, for example through synchronous firing [6]. Independent-neuron codes discard this information and assume that the neurons respond statistically independently of each other. Empirical data supports the independent-spike hypothesis and is often compatible with independent-neuron codes [6]. This work assumes such independence, which allows describing neural activity in terms of firing rates.

2.2.1 Firing rates

Because synaptic transmission is triggered by action potentials and these are highly stereotyped, neural activity is well described by the spike trains of the neurons in a network [6]. Spike trains often exhibit large trial-to-trial variability, i.e., they can look very different in repeated trials with the same stimulus. Trial-to-trial variability is uninformative of the stimulus and thus often considered as noise, although it could also represent internal state or other, uncontrolled variables [74]. Together with the discrete nature of spike trains, this makes them hard to interpret. One therefore often averages spike trains to obtain continuous firing rates. Firing rates can be defined as averages over trials, neurons, or time. The question is whether these firing rates are a good approximation of the underlying spike trains, in the sense that they capture most of the information about the stimulus [6].

The coding perspective is primarily an outside view that connects stimulus and activity without asking how the network can generate or use the activity. Sec. 2.3.1 introduces networks of rate neurons that dynamically interact through their continuous firing rates.

2.2.2 Receptive fields

The receptive field (RF) of a neuron is the region in sensory space that it responds to [6, 7, 68]. Furthermore, RFs are patterned, describing the structure of sensory stimuli within the RF region that drive a neuron [6, 7, 68]. As an example, many neurons in the primary visual cortex (V1) respond best to oriented stimuli in a small region of visual space that have separated darker and brighter (ON- and OFF-) subregions [68]. The responses of these *simple cells* can be described by a linear RF, which is the linear filter that best explains how a neuron responds to sensory input [7, 75]: Inputs that are aligned with the RF evoke strong responses, while misaligned or non-overlapping inputs evoke weak or no responses. To predict the activity of a neuron, a sensory pattern is first projected onto its RF. The resulting value is then nonlinearly transformed into an estimate of its firing rate, based on which spike times can be randomly drawn [7]. Chapter 4 studies how recurrent neural networks may give rise to neurons with linear RFs in an efficient way.

The relationship between sensory input and neuronal activity can be more complex than for simple cells in V1. As an example, the responses of *complex cells* in V1 cannot be described by linear RFs; for example, they can respond with equally high activity to a stimulus with inverted brightness [76]. In general, the RFs of neurons further along the hierarchy of the visual system become more complex and broader [68, 77–80].

Because stimuli can be complex, and because neuronal responses can depend on stimulus features that are not linearly available, it is often useful to characterize stimuli in terms of a single or few relevant stimulus attributes. The neuronal response to stimuli as a function of such attributes is called a tuning curve [6]. For example, V1 neurons can have responses that are modulated by, or selective for, stimulus orientation, direction of movement, color, or input from the left versus right eye [68]. For neurons whose responses are well predicted by their RFs, the tuning curves follow from the RF. However, also neurons for which this is not the case, like complex cells, can be described in terms of tuning curves [68].

2.2.3 Predictive coding

Chapter 4 studies a 'cooperative coding' scheme that minimizes the number of synapses and gives rise to net excitation between similarly-tuned neurons. Like-to-like excitation is found in experiments [43–47, 81], but it is unclear why neural networks evolved that way. The role of this excitation is especially unclear in light of another, 'predictive' coding scheme [48, 49, 82] in which like-to-like *inhibition* naturally arises to decorrelate and coordinate neuronal spiking. The objective there is to efficiently encode continuous signals in a neuronal population, while minimizing the number of spikes or the rate activity. Through their competitive inhibition, similarly-tuned neurons prevent redundant, surplus spiking, thus minimizing fluctuations in the readout and increasing its precision. The described benefits for like-to-like inhibition, together with the fact that excitation would have the opposite effect, add to the challenge of understanding the role of the observed excitation.

The predictive coding scheme can be derived starting from a population of spiking neurons with otherwise unspecified dynamics [48]. The dynamics become specified when the population is optimized to approximate a linear dynamical system by a linear readout of the exponentially filtered spike trains of the network. The target system of dynamical variables $\mathbf{x}$ is described by $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{c}(t)$, where $\mathbf{A}$ is the state transition matrix and $\mathbf{c}(t)$ external input. The estimate of the neuronal network, $\hat{\mathbf{x}}$, evolves as $\dot{\hat{\mathbf{x}}} = -\lambda_d \hat{\mathbf{x}} + \mathbf{\Gamma}\mathbf{o}(t)$, with λ_d the inverse of the filtering time constant, $\mathbf{\Gamma}$ the readout matrix, and $\mathbf{o}(t)$ the vector of spike trains. The readout $\mathbf{\Gamma}$ is, unusually, not optimized but given. The estimate $\hat{\mathbf{x}}$ can also be expressed $\hat{\mathbf{x}}(t) = \mathbf{\Gamma}\mathbf{r}(t)$ in terms of firing rates $\mathbf{r}$ defined by $\dot{\mathbf{r}} = -\lambda_d \mathbf{r} + \mathbf{o}(t)$. The spike times $\mathbf{o}(t)$ of the neurons are chosen such that they greedily minimize the instantaneous loss (here for simplicity without regularizations that explicitly penalize neural activity)

$$L(t) = \|\mathbf{x}(t) - \hat{\mathbf{x}}(t)\|_2^2, \tag{2.1}$$

meaning that any neuron spikes whenever the loss at that point in time is lower with a spike than without a spike [48]. This inequality can be reformulated as neuron i spiking whenever

$$V_i(t) > T_i, \tag{2.2}$$

where V_i and T_i are interpreted as the neuron's membrane potential and spiking threshold, respectively,

and turn out as [48]

$$V_i(t) = \mathbf{\Gamma}_{\cdot i}^T(\mathbf{x}(t) - \hat{\mathbf{x}}(t)), \qquad T_i = \left\|\mathbf{\Gamma}_{\cdot i}\right\|_2^2/2. \tag{2.3}$$

The membrane potential of any neuron i thus tracks the instantaneous prediction error $\mathbf{x}(t) - \hat{\mathbf{x}}(t)$ projected onto its readout weights $\mathbf{\Gamma}_{\cdot i}$.

The spiking condition can be understood as follows [48]: A spike of neuron i changes the prediction error by $-\mathbf{\Gamma}_{\cdot i}$. As soon as the prediction error in the direction of $\mathbf{\Gamma}_{\cdot i}$ (meaning a projection onto $\mathbf{\Gamma}_{\cdot i}/\left\|\mathbf{\Gamma}_{\cdot i}\right\|_2$) is larger than $\left\|\mathbf{\Gamma}_{\cdot i}\right\|_2/2$, a spike would lower it and the neuron should spike. The spiking threshold is hence $\left\|\mathbf{\Gamma}_{\cdot i}\right\|_2^2/2$.

Because neurons spike whenever they can lower the prediction error, but there are no spikes that would increase it, the code is very efficient in terms of the number of spikes. It is also efficient in terms of network size N: The readout $\mathbf{\Gamma}$, which determines the maximum prediction error, should scale with $1/N$ to maintain neuronal firing rates [48]. In contrast, for a rate coding network with Poissonian spiking, the prediction error scales only with $1/\sqrt{N}$ [48].

The dynamics of the membrane potential can be obtained by differentiating $V_i(t)$ and using $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x}+\mathbf{c}(t)$, $\dot{\hat{\mathbf{x}}} = -\lambda_d\hat{\mathbf{x}} + \mathbf{\Gamma}\mathbf{o}(t)$, and $\hat{\mathbf{x}} = \mathbf{\Gamma}\mathbf{r}(t)$, resulting in [48]

$$\dot{V}_i(t) = \mathbf{\Gamma}_{\cdot i}^T\left(\mathbf{A}\mathbf{x}(t) + \mathbf{c}(t) + \lambda_d\mathbf{\Gamma}\mathbf{r}(t) - \mathbf{\Gamma}\mathbf{o}(t)\right). \tag{2.4}$$

To reexpress $\mathbf{x}$ in terms of the membrane potential, one assumes that $\mathbf{\Gamma}$ has a left pseudoinverse $\mathbf{L} = (\mathbf{\Gamma}\mathbf{\Gamma}^T)^{-1}\mathbf{\Gamma}$. Multiplying Eq. (2.3) from the left with $\mathbf{L}$ yields $\mathbf{x}(t) = \mathbf{L}\mathbf{V}(t) + \hat{\mathbf{x}}(t)$. Inserting this into Eq. (2.4), neglecting terms that vanish in the limit of large networks and adding a generic leak term $-\lambda_V\mathbf{V}(t)$ by hand, the membrane dynamics of the network are [48]

$$\dot{\mathbf{V}}(t) = -\lambda_V\mathbf{V}(t) + \frac{1}{\lambda_d}\mathbf{\Omega}^{\mathrm{s}}\mathbf{r}(t) - \mathbf{\Omega}^{\mathrm{f}}\mathbf{o}(t) + \mathbf{\Gamma}^T\mathbf{c}(t) \tag{2.5}$$

with slow connections

$$\mathbf{\Omega}^{\mathrm{s}} = \mathbf{\Gamma}^T(\mathbf{A} + \lambda_d\mathbf{I})\mathbf{\Gamma} \tag{2.6}$$

and fast connections

$$\mathbf{\Omega}^{\mathrm{f}} = \mathbf{\Gamma}^T\mathbf{\Gamma}. \tag{2.7}$$

Together with the spiking mechanism, the derived neuronal dynamics are those of leaky integrate and fire (LIF) neurons [7] with current-based interactions.

The network connectivity is comprised of slow rate interactions ($\propto \mathbf{r}$) and fast spike interactions ($\propto \mathbf{o}$). Both are determined by the readout weights $\mathbf{\Gamma}$, while the slow connectivity also depends on $\mathbf{A}$. Consequently, the slow connections determine the population dynamics mirroring the dynamical system described by $\mathbf{A}$. For slow or even integrator dynamics ($\mathbf{A} \approx 0$), the slow connectivity $\mathbf{\Omega}^{\mathrm{s}}_{ij} \approx \lambda_d\mathbf{\Gamma}_{\cdot i}^T\mathbf{\Gamma}_{\cdot j}$ between two neurons is proportional to the scalar product of their readouts. The interaction is thus excitatory for neurons that contribute similarly to the readout. In contrast, the fast connections $-\mathbf{\Omega}^{\mathrm{f}}$ implement fast (near instantaneous) inhibition between similarly-tuned neurons, where the diagonal elements can be interpreted as a spike reset [48].

For a slow dynamical system, slow and fast interactions thus provide (tightly) balanced excitation and inhibition [48]. However, if the task was just to quickly track an unpredictable input stimulus, and we acknowledge that the readout can only do so with a time constant set by λ_d, this would correspond to $\mathbf{A} = -\lambda_d \mathbf{I}$ and therefore $\mathbf{\Omega}^{\mathrm{s}} = 0$. For tracking fast stimuli with minimal integration, network connectivity is thus not balanced but characterized by strong like-to-like inhibition.

Whenever a neuron spikes, this increases the network output and thus decreases the prediction error by $\mathbf{\Gamma}_{\cdot i}$. As the neurons encode the prediction error in their membrane potential (Eq. (2.3)), this change needs to be communicated to the other neurons. From Eq. (2.3), the change in V_i upon a spike of neuron j should be $\mathbf{\Gamma}_{\cdot i}^T(-\mathbf{\Gamma}_{\cdot j}) = -(\mathbf{\Gamma}^T\mathbf{\Gamma})_{ij}$, explaining $\mathbf{\Omega}^{\mathrm{f}}$ (Eq. (2.7)).

One implication of the need for instant broadcasting of prediction errors is that neuronal connectivity must be very dense: any two neurons with non-orthogonal readout kernels share two fast connections. This is in contrast to the sparse connectivity in the brain [6] and to the cooperative coding scheme developed in Chapter 4.

2.3 Neural networks as dynamical systems

Describing neural networks as dynamical systems makes them available to a lot of analysis tools. As these tools do not readily apply to spiking neurons, it is useful to describe neuronal activity using continuous firing rates. Such networks of rate units are reviewed in Sec. 2.3.1 and their linearized dynamics given in Sec. 2.3.2. A measure for the effective dimensionality of neuronal activity, the participation ratio (PR), is motivated in Sec. 2.3.3. Sec. 2.3.4 describes a relevant cortical dynamical regime, the balanced state, which can lead to fast and linear population responses to external inputs. Balance between excitation and inhibition also causes an effect called balanced amplification, reviewed in Sec. 2.3.5.

2.3.1 Rate networks

Neurons interact with each other via spike trains. For independent-spike and independent-neuron codes one can define firing rates that capture most of the information contained in the spiking network activity [6], c.f. Sec. 2.2.1. To define networks of rate neurons, one has to further assume that correlations and precise spike patterns that are not captured in the firing rates have little impact on the rate dynamics [6]. In other words, firing rate descriptions have to be not only informative, but also sufficient to explain and predict later rate dynamics.

One difficulty is that even for uncorrelated inputs, the spiking mechanism introduces correlations between inputs and outputs, and between neurons that receive shared inputs [74]. This is particularly the case in the fluctuation-driven regime, in which input fluctuations contribute strongly to the neurons crossing their spiking thresholds. In this regime, neurons have irregular, Poisson-like spike trains, and are especially sensitive to correlated input fluctuations [74]. In multilayer- or recurrent networks, this can lead to correlated firing inconsistent with the assumptions of rate coding. The problem can be alleviated by sparse connectivity, private noise, or cancellation of correlations as when inhibition tracks excitation [74, 83].

It is however questionable whether there is an *exact* mapping between spiking and rate dynamics, in the sense that evolving a spiking network and then transforming spike trains into firing rates yields the same as first obtaining the initial firing rates and then evolving the corresponding rate network [74]. Still, a *good* correspondence is often possible. This is evidenced by rate models that describe and predict experimental neuronal data [84], explain observed neuronal dynamics [85–87], behave like corresponding spiking networks [88], or provide correspondences between rate units and neurons of different subtypes [89–91].

On an individual trial, the spike train of a neuron, even if convolved with a short temporal filter, can often look very different from a smooth, continuous rate, as for example defined by the neuron's trial-averaged firing rate. This is especially true at low firing rates and might make a rate description seem unrealistic. However, a neuron typically receives thousands of inputs [68]. This means that, compared to its output, its total synaptic input has much less trial-to-trial variability and can be better described as a continuous quantity [6]. In networks with low-dimensional neuronal activity, both activity and inputs can then be described as a linear combination of few latent 'factors', which represent the network activity along directions in neuron space that capture most of the neuronal variability [88]. This allows to replace a large network of sparsely connected spiking neurons by a smaller network of densely connected rate neurons [6]. Conceptually, a single model unit represents the average response of multiple neurons with similar responses [88], although any neuron might contribute to multiple factors and be represented by multiple rate units.

Because there is no exact rate description of spiking networks [74], the dynamic equations governing rate networks are to a certain degree *ad hoc* [7]. In this work, the state of the ith rate neuron is described by a single continuous variable x_i, which is a low-pass filtered version of the neuron's total input,

$$\tau \dot{x}_i(t) = -x_i(t) + \sum_{j \neq i} W_{ij} r_j(t) + I_i(t). \tag{2.8}$$

Here τ is the filtering time constant, I_i is the external input to neuron i and $\sum_{j \neq i} W_{ij} r_j(t)$ its recurrent input from other neurons that fire with rates $\mathbf{r}$, summed linearly. These rates do not stem from an explicitly modelled spiking mechanism; instead a nonlinearity g translates the 'pre-activations'$\mathbf{x}$ into continuous rates $\mathbf{r}$,

$$r_i(t) = g(x_i(t)). \tag{2.9}$$

The nonlinearity $g(I_i^c)$ describes the firing rate in response to stationary total input I_i^c, which in that case equals x_i. Eqs. (2.8,2.9) thus model the instantaneous firing rate of a neuron as its stationary input-to-firing rate relation applied to a low-pass filtered version of its total input. The time constant of the rate dynamics, τ, is often chosen as the membrane time constant of the modelled spiking neurons [7], but can in general be different, and often much shorter [6].

The rate description is conceptually closest to a corresponding spiking neural network if rate units model the mean firing rates of non-overlapping, homogeneous neuron populations. The weight matrix $\mathbf{W}$ then combines both the connectivity and the typical synaptic strength between populations and the nonlinearity can be inferred from the gain functions of single neurons [7]. The rate unit can also be reinterpreted as a model for the trial-averaged firing rate of a single stochastically firing neuron, which in a homogeneous population agrees with the population-average [7]. The correspondence between rate units and specific spiking neurons becomes tangled when rate units reflect latent factors in the spiking activity [88], meaning that each neuron can contribute to many factors simultaneously.

2.3.2 Linear dynamical systems

Chapters 3,4 mainly employ linear rate networks, which are more amenable to analytical exploration and understanding. This simplification sacrifices some computational power of the neural networks, as linear networks can only implement linear functions, emphasizing the importance of the neuronal nonlinearity for computations. However, linear networks are more amenable to analytical exploration and understanding, and still expressive enough for the phenomena studied in this thesis, as argued in the Chapter 5.

For a linear network, $g(\mathbf{x})$ is simply the identity function. The network dynamics, Eqs. (2.8,2.9), can then be written as

$$\tau \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{I}(t), \tag{2.10}$$

with $\mathbf{A} = -\mathbb{1} + \mathbf{W}$. Because $\mathbf{x}$ appears only linearly, the full solution of Eq. (2.10) is the sum of a particular solution of Eq. (2.10) and a general solution of the homogeneous problem

$$\tau \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t), \tag{2.11}$$

with coefficients fully determined by the initial conditions [92]. For a physical system with real observables $\mathbf{x}$ such as a neural network, $\mathbf{A}$ and $\mathbf{I}$ are real, while the solutions $\mathbf{x}^{(k)}$ and coefficients a_k can generally be complex. However, from Eq. (2.10) it is evident that the solution for a given real initial state $\mathbf{x}(t_0)$ will stay real.

Following from the superposition principle, any solution $\mathbf{x}$ of Eq. (2.11) can be written as a sum of 'basis functions', and there are $N = \dim\ \mathbf{x}$ of them. This can be easily seen for the case where $A = UDU^{-1}$ has a diagonal representation D in another basis, specified through the change-of-basis matrix U. The eigenvalues of A are then the diagonal entries $\Omega_i = D_{ii}$ of D. Expressing the dynamical variables in the new basis, $\mathbf{y} = U^{-1}\mathbf{x}$, the system of linear differential equations decouples into N one-dimensional differential equations,

$$\tau\dot{\mathbf{y}}(t) = U^{-1}\mathbf{A}U\mathbf{y}(t) = \mathbf{T}\mathbf{y}(t) \qquad \Rightarrow \qquad \tau\dot{y}_i(t) = \Omega_i y_i(t), \tag{2.12}$$

which are solved by $y_i(t) = y_i(0)\exp(\Omega_i t/\tau)$. For a real square matrix, the eigenvalues $\Omega_i = \lambda_i + \mathrm{i}\omega_i$ are generally complex, with non-real eigenvalues appearing in complex conjugate pairs [93]. Negative (positive) λ_i describe an exponential decay (growth) of network activity, while nonzero ω_i describe oscillations.

It would require fine-tuning of the individual synaptic strengths or the entries of A, respectively, to achieve a matrix with repeating eigenvalues. For example, in the large N limit, the eigenvalues of a real matrix with iid. entries with mean zero becomes uniformly distributed on the (scaled) complex unit disk, with zero probability mass for equality of two eigenvalues [94, 95]. There is no reason to assume such fine-tuning for biological neural networks. On the contrary, they are often modeled as random [60] or containing a random component [88]. The assumption of a full set of distinct eigenvalues is thus typically satisfied.

However, the set of corresponding eigenvectors is in general not orthogonal. In this case the change-of-basis matrix $\mathbf{U}$ is non-unitary and $\mathbf{A}$ non-normal [93]. The implication is that Eq. (2.12) decouples then only for neuronal activity along eigenvectors which form a non-orthogonal basis, but not if an orthogonal basis is used. One effect from such connectivity is balanced amplification [51], discussed in Sec. 2.3.5.

The general solution to Eq. (2.10) involves a convolution of a matrix exponential of $\mathbf{A}$ and the inhomogeneous external input $\mathbf{I}$, in addition to the homogeneous solution [92],

$$\mathbf{x}(t) = \mathrm{e}^{\mathbf{A}\frac{t}{\tau}}\mathbf{x}(0) + \int_0^t \mathrm{d}s\ \frac{1}{\tau}\mathrm{e}^{\mathbf{A}\frac{t-s}{\tau}} I(s). \tag{2.13}$$

2.3.3 Dimensionality of neuronal activity

Neuronal activity (let's think of trial-averaged rates) can be naturally expressed as a vector $\mathbf{r}(t)$, with each component representing the activity of one neuron. For a network of N neurons, this activity vector is thus N-dimensional. Across conditions and/or time the neural network assumes different such activity states. Together these form the set of actually visited states, which typically occupies a small subset of the available, high-dimensional state space. This is compatible with the idea that neural activity represents relatively lower-dimensional systems that underlie some constraints [38], or, respectively, that the rates of different neurons are not independent but correlated [39].[1] The effectively low-dimensional

[1] This statement seems to contradict the assumption of the balanced state that neuronal firing is uncorrelated. However, the balanced state requires only that the 'spike' or 'noise correlations', i.e., the correlations of fast fluctuations around the more

nature of the data lessens the curse of dimensionality [96] and is crucial for the performance of WP and NP, analyzed in publication 1 [1], Chapter 3.

The covariances between the activities of different neurons can be captured in their covariance matrix $S_{ij} = \mathrm{Cov}[r_i, r_j]$, which is positive-semidefinite. The spectral theorem states that S, as a real symmetric matrix, has a diagonal representation in an orthonormal basis consisting of its eigenvectors [93]. Principal component analysis (PCA) defines these eigenvectors as the principal components, with the corresponding eigenvalues as principal values [97]. The principal values thereby equal the variance of the data projected onto their associated principal components, so that a component with a large principal value captures a lot of the data's variance [97]. This motivates the use of PCA as a dimensionality-reduction technique: by projecting the data onto the subspace spanned by the principal components with the largest principal values, few components suffice to explain a large part of the total variance [97].

Chapter 3 uses a measure for the effective dimensionality of neural activity called participation ratio (PR) [38]. Effective dimensionality is a descriptive measure of the full neural data that does not distinguish between signal and noise, or important and unimportant signal dimensions [38]. This is in contrast to measures of intrinsic dimensionality, which measure the number of latent variables needed to describe the signal part of the data [38, 98]. As a linear measure, the PR can be higher than the intrinsic dimension if the data lies on a nonlinear manifold that is embedded in a higher dimensional space. It can in general also be lower, if the neural manifold is linear and the latent variables correlated [38].

The concept of PR builds on the spectral entropy of the data, which can be calculated from its principal values [38]. The normalized eigenvalues $p_i = \lambda_i / \sum_j \lambda_j$ of the covariance matrix measure the relative contribution of each principal component to the total variance. As they sum to one, they can be interpreted as (pseudo) probabilities. The PR is defined as the equivalent number of orthogonal dimensions with equal variance that have the same quadratic spectral entropy (which is the Rényi entropy of order 2) as the original data [38]:

$$\underbrace{-\ln\left(\sum_{i=1}^{N} p_i^2\right)}_{\text{original spectral entropy}} = \underbrace{-\ln\left(\mathrm{PR}\left(\frac{1}{\mathrm{PR}}\right)^2\right)}_{\text{equivalent spectral entropy}}, \tag{2.14}$$

where $1/\mathrm{PR}$ are the (normalized) principal values of the first PR principal components of the equivalent system. The equation is solved by

$$\mathrm{PR} = \frac{1}{\sum_{i=1}^{N} p_i^2} = \frac{\left(\sum_{j=1}^{N} \lambda_i\right)^2}{\sum_{i=1}^{N} \lambda_i^2}. \tag{2.15}$$

The definition aligns with the intuition that for a high effective dimensionality the data variance should be spread across many directions. For the case where the variance is equally distributed over N_{eff} directions, the PR equals N_{eff}. In particular, $\mathrm{PR} = 1$ if all neurons are perfectly correlated and $\mathrm{PR} = N$ if they have iid. activity, which is the largest possible value.

slowly varying firing rates, are small.

2.3.4 Balanced state

Chapter 4 develops a cooperative coding scheme [2] that is analyzed for a linear rate network as described in Secs. 2.3.1,2.3.2. The network dynamics turn out to be slow when only excitatory connections are present. One result of publication 2 [2], Chapter 4, is that the dynamics can be greatly sped up when relatively strong excitatory inputs are balanced by inhibition that briefly lags behind excitation. The mechanism behind this speed-up is essentially balanced amplification [51], as briefly described in Sec. 2.3.5. This subsection gives a brief introduction to the balanced state, which is a regime of network dynamics that features the cancellation of the largest part of strong excitation and inhibition required for the speed-up mechanism. Experimental evidence for lagged, balancing inhibition is reviewed in the discussion of publication 2 [2], Chapter 4.

Depending on the brain region and activity state, neural networks can operate in different dynamical regimes. One such regime that is often observed in cortical areas is asynchronous, irregular firing [62]. Irregular firing pertains to a high variability of interspike intervals, as opposed to spikes that occur in a more regular 'rhythm'. Asynchronous firing means that the firing of different neurons is not or only weakly correlated. The balanced state [60, 61] offers an explanation and a mechanism for both features of neural activity.

Consider a single neuron that receives Gaussian input with a certain mean and variance, approximating the summed effect of many uncorrelated synaptic inputs. Depending on the mean and variance, spiking will be mean- or fluctuation-driven. For super-threshold mean and small variance, the neuron approaches its spiking threshold rather deterministically. Because the input noise is weak, the time from reset to the next spike will vary only little so that the neuron spikes regularly. On the other hand, if the input has sub- or peri-threshold mean but large variance, the neuron will (only) spike as a result of input fluctuations. As these are random, also the spike times and their intervals are random so that the neuron spikes irregularly [60, 83].

Consider now two irregularly spiking neurons in the fluctuation-driven regime. If their inputs are correlated, then so will be their outputs [74]. This becomes evident for the extreme case where they receive the same input. Asynchronous firing thus requires that neurons receive uncorrelated or only weakly correlated input [74]. This is typically justified by high network sparsity, so that neurons have little shared input [83].

How can a network achieve a state in which neuronal spiking is fluctuation-driven? For typical parameters, this is not possible with excitation alone [99]. The reason is that, for many weak inputs, a large variance implies that the mean is also large. This is not the case when inputs can also be inhibitory: while each (uncorrelated) input increases the variance of the total input, it can lower its mean if it has opposite sign. In the balanced state excitation and inhibition are correlated such that their sum is (much) smaller than either of them [62]. The balance and correlation of excitation and inhibition has differing strengths in different variants of the balanced state.

A classic result [60] concerns balanced networks in which synaptic strengths scale as $\sim 1/\sqrt{K}$ with the number of presynaptic inputs K per neuron, such that only $\sim \sqrt{K}$ excitatory synaptic inputs are required to elicit a spike. Such balance is termed 'tight balance' in ref. [62], meaning that the net input is very small compared to the individual excitatory and inhibitory currents, while it is termed 'loose balance' in refs. [49, 100], meaning that fast fluctuations of excitation and inhibition do not cancel on short timescales. Consider two recurrently coupled homogeneous populations of excitatory and inhibitory neurons with rates ν_E and ν_I, respectively, that receive external input ν_X. For simplicity, assume that the neuronal parameters, such as integration time constant τ, as well as the numbers and strengths of

incoming connections are the same for both populations. Consequently, they fire at the same average rate $\nu = \nu_E = \nu_I$. Let the excitatory synaptic strengths be J_{EE} and J_{IE}, and inhibitory and external strengths scaled versions of the excitatory weights, i.e., $J_{AB} = g_B J_{AE}$ for $A \in E, I$ and $B \in I, X$. Then, if there are γ times as many inhibitory as excitatory inputs, the mean and variance of the total input that both populations receive are [59, 60, 101]

$$\mu = \tau JK \left(g_X \nu_X - (\gamma g_I - 1)\nu\right) \overset{!}{\approx} O(\theta), \tag{2.16}$$

$$\sigma^2 = \tau J^2 K \left(g_X^2 \nu_X + (\gamma g_I^2 + 1)\nu\right) \overset{!}{\approx} O(\theta^2), \tag{2.17}$$

where θ is the spiking threshold and J and K are the excitatory strength and indegree, respectively (versus $g_I J$ and γK for inhibitory inputs).

Importantly, Eqs. (2.16,2.17) describe a 'loosely balanced' state that assumes that individual inputs are independent [49]. To achieve this, neurons are sparsely connected ($K \ll N$) to minimize shared input. Independent input here implies that the fast fluctuations of the total excitation and inhibition a cell receives are uncorrelated. However, the mean excitation and inhibition, which co-evolve on a slower timescale, are correlated.

The network is inhibition-dominated, i.e., $\gamma g_I > 1$, which prevents excitation from causing saturated firing at high rates [101]. In the balanced state, both the mean and standard deviation are on the order of the spiking threshold θ: were it otherwise, neurons would be either completely silenced or maximally driven [60]. This balance is achieved dynamically by the adjustment of the neurons' firing rates ν, consistent with the negative feedback loop from dominating inhibition (the $-(\gamma g_I - 1)\nu$ term in Eq. (2.16), with ν in turn increasing with μ). Remarkably, this balance arises as a dynamic phenomenon in a wide parameter range without the need for fine-tuning [60, 101].

Given that the mean input remains comparable to the firing threshold, and that synaptic interactions are strong (large JK), the external and recurrent inputs in Eq. (2.16) cancel in good approximation. This reveals (by setting the RHS approximately to θ) that the network response can be well approximated as a (threshold-) linear function of the external input [59, 60],

$$\nu \approx \left[\frac{g_X \nu_X - \nu_{th}}{\gamma g_I - 1}\right]_+, \tag{2.18}$$

where $\nu_{th} \approx \theta/\tau JK$ and $[\cdots]_+$ denotes rectification. This linearity on the population level stands in contrast to the highly nonlinear responses of single neurons [60] (see also ref. [102]).

The predictive coding scheme [48, 49] described in Sec. 2.2.3 assumes even stronger balance: synaptic strengths are $O(1)$ and do not decrease with indegree K. This 'tight balance' leads to correlated fluctuations of excitation and inhibition, so that the E/I balance holds even on short time scales, with inhibition briefly lagging behind excitation [49, 100]. Ref. [62] argues that cortex is instead in a 'loosely balanced' regime, which they define as one in which the net input is not much smaller than the individual excitatory or inhibitory input. Such weaker balance might have the computational advantage of generating nonlinear population responses [59].

.

2.3.5 Balanced amplification

Neural networks can selectively amplify patterns of activity; for example, spontaneous activity in cat V1, presumably the result of the amplification of specific patterns present in random input fluctuations, is non-random but resembles stimulus-evoked activity [51, 103]. One possible mechanism is the amplification through appropriate large feedforward weights. Another mechanism leverages recurrent excitation to form a positive-feedback loop in which an activity pattern reinforces itself. The amplified pattern corresponds to an eigenvector of the weight matrix with a positive eigenvalue, see Eq. (2.12), which compensates for the intrinsic decay of network activity, Eq. (2.8). Such 'Hebbian amplification' thus slows the decay of activity in the pattern and thereby the network dynamics.

Balanced amplification, a concept introduced by Murphy and Miller [51], describes selective amplification in a recurrent network without significant slowing of dynamics. The network thereby implements in its recurrent weights an effective feedforward connectivity between activity modes. Balanced amplification is present in any (sufficiently dense) recurrent network that satisfies Dale's law and becomes significant when strong excitation is stabilized by inhibition [51], as is often the case in cortex [104, 105]. The cooperatively coding networks of publication 2 [2], from Chapter 4 are augmented with delayed, balancing inhibition to implement balanced amplification to speed up their dynamics.

To understand balanced amplification, recall that the neural weight matrix typically has a diagonal representation in some basis, only not in an orthogonal one (c.f. Sec. 2.3.1). Only normal matrices, defined as those commuting with their conjugate transpose, $AA^\dagger = A^\dagger A$, have an orthogonal basis in which they are diagonal [93]. It is easy to see that the weight matrix of a network of excitatory and inhibitory neurons does not commute with its transpose,

$$\underbrace{\begin{pmatrix}(+) & (-)\\ (+) & (-)\end{pmatrix}}_{\mathbf{W}}\underbrace{\begin{pmatrix}(+) & (+)\\ (-) & (-)\end{pmatrix}}_{\mathbf{W}^T} = \begin{pmatrix}(+) & (+)\\ (+) & (+)\end{pmatrix} \neq \begin{pmatrix}(+) & (-)\\ (-) & (+)\end{pmatrix} = \underbrace{\begin{pmatrix}(+) & (+)\\ (-) & (-)\end{pmatrix}}_{\mathbf{W}^T}\underbrace{\begin{pmatrix}(+) & (-)\\ (+) & (-)\end{pmatrix}}_{\mathbf{W}}, \tag{2.19}$$

where $(\pm)$ stands for submatrices with non-negative/non-positive entries, respectively. Consequently, weight matrices of networks that respect Dale's law are non-normal.

It is instructive to study the case of two coupled populations of excitatory and inhibitory neurons receiving the same input (like the example from [60] in Sec. 2.3.4) as given in [51], described by weights

$$\mathbf{W} = \begin{pmatrix} w & -kw \\ w & -kw \end{pmatrix}, \tag{2.20}$$

with $k > 1$ signifying inhibition dominance. The eigenvalues of $\mathbf{W}$ are $-w_+ = -w(k-1)$ and 0, corresponding to the eigenvectors $\frac{1}{\sqrt{2}}\binom{1}{1}$ and $\frac{1}{\sqrt{1+k}}\binom{k}{1}$. Clearly, these are non-orthogonal. Moreover, for $k \to 1^+$, their difference goes to zero as the second eigenvector aligns with the first. This implies that even very weak activity along $\frac{1}{\sqrt{2}}\binom{1}{-1}$, which is perpendicular to the first eigenvector and almost perpendicular to the second one, can only be expressed as the difference in the eigenmodes with very large coefficients. As the eigenmodes decay at different rates, their activities no longer cancel along the direction of the first eigenvector. Consequently, neural activity acquires a large component parallel to the first eigenvector and perpendicular to the initial activity, a phenomenon known as transient amplification.

The above reasoning motivates the use of the orthonormal vectors $\mathbf{r}_+ = \frac{1}{\sqrt{2}}\binom{1}{1}$ and $\mathbf{r}_- = \frac{1}{\sqrt{2}}\binom{1}{-1}$ as basis vectors, which [51] call the sum and difference mode. Note that only the sum mode $\mathbf{r}_+$ is an

eigenvector. The change of basis reveals that the weight matrix

$$\tilde{\mathbf{W}} = \mathbf{O}\mathbf{W}\mathbf{O}^T = \begin{pmatrix} -w_+ & w_{FF} \\ 0 & 0 \end{pmatrix}, \qquad \mathbf{O} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \tag{2.21}$$

assumes an upper-diagonal form [51]; in other words it has a hidden feedforward structure from the difference mode to the sum mode with feedforward weight $w_{FF} = w(1 + k)$. In more applied terms, the sum mode corresponds to equal excitation and inhibition, which decays faster than in an uncoupled network due to the net inhibition. The difference mode corresponds to a surplus of excitatory activity not (yet) balanced by inhibition. It contributes (for strong balance, i.e., large w and k: strongly) to both excitation and inhibition, thus driving the sum mode. In summary, small differences in excitation and inhibition are amplified and yield large increases in both excitation and inhibition [51].

This effect is not the result of the particular choice for the weight matrix in Eq. (2.20). Rather, Schur's lemma states that any (complex) square matrix has an orthonormal basis in which it assumes an upper-diagonal form [93]. This so-called Schur basis can be found by first finding a basis in which $\mathbf{W}$ is upper-triangular, and then orthonormalizing the basis vectors with the Gram-Schmidt procedure, which does not change the upper-triangular form [93]. If the matrix is non-normal, then it cannot be unitarily diagonalized and its upper-diagonal form contains at least one nonzero off-diagonal, i.e., feedforward, entry.

2.4 Fluctuation-driven reinforcement learning

2.4.1 Noise in neural systems

In the typical view of neuronal networks, synaptic connectivity is the main determinant of network function [3, 4]. Accordingly, synaptic plasticity generally modifies network function and represents learning. This learning depends on current and past experiences and, from the perspective of a synapse, happens in the context of pre- and postsynaptic activity histories [19]. There are many suggested forms of activity-dependent plasticity, which depends on these histories [7].

From a single synapse's perspective, determining how to modify its strength to improve global network function is a hard task. One part of the task is termed the credit assignment problem, i.e., determining how strongly and in what way a synapse contributed to the network dynamics and influenced possible rewards. In machine learning, one often solves this general problem in artificial neural networks through backpropagation [106]. For biological networks, it is exacerbated by the fact that synapses have, in contrast to algorithms that run on a computer, access only to locally available information. Even for attempts to formulate biologically plausible versions of backpropagation [35], this local information crucially involves pre- and postsynaptic activity histories. It is thus plausible and reasonable that synaptic plasticity should depend on neuronal activity, and many experiments demonstrate such dependence [7, 68].

A consequence of the activity dependence of synaptic plasticity, and also of the view that plasticity represents learning and coordinated changes of network function, is that synapses should remain stable in the absence of activity [19]. This ability of synapses to retain their properties over behaviorally relevant time scales is termed synaptic tenacity [19]. Conserving synaptic properties, however, poses considerable challenges to the maintaining biological machinery: synaptic proteins exhibit significant dynamics and life cycles with time scales of hours and less, orders of magnitude smaller than the lifetimes

of most CNS synapses [17, 19, 107]. It is thus reasonable to expect that synaptic strengths will, with time, unavoidably experience some random 'drift'. Such learning-unrelated synaptic modifications are part of so-called activity-independent plasticity, which subsumes plasticity that does not depend on activity in an explicit way, and is presumably learning-unrelated. Publication 1 [1], Chapter 3, argues that a part of activity-independent plasticity could actually represent learning, as well as exploratory weight fluctuations that enable it.

There is ample evidence for activity-independent plasticity in the CNS that is comparable in magnitude with activity-dependent plasticity [16–19]. Even in experiments that explicitly measure activity-dependent plasticity under highly controlled conditions, deterministic effects are obscured by substantial 'noise' [108]. Size changes in pairs of synapses that share the same axon and dendrite only weakly correlate, with only 40 % of remodeling attributed to activity-dependent plasticity [109]. Further, synapses in the intact brain are not static, but experience continuous strength fluctuations [19]. Measuring the volumes of dendritic spines, which are a proxy for synaptic strength [110], reveals not only size fluctuations, but also structural plasticity: New spines appear as others disappear. Transient spines that last at most a few days are thereby thin and small, and can make up about 20 to 60 % of spines [16, 111]. Persistent spines that last more than a week are relatively thick and likely to survive months and possibly longer, although also 'persistent' spines can be lost [16, 111].

Experiments in cortical [107] and hippocampal [17] primary cultures show that silencing of network activity decreases spine size fluctuations, consistent with the presence of activity-dependent plasticity. However, synaptic drift and structural plasticity continue at a rate comparable with activity-dependent plasticity [17, 107]. Similarly, inducing (re-)learning by trimming a subset of whiskers of behaving mice increased spine turnover rates selectively in receptive principal cells in barrel cortex, but only by about 50 % [16]. Spontaneous fluctuations of synaptic strength occur also in inhibitory synapses, although these appear to drift more slowly [18]. In contrast to excitatory synapses, suppression of network activity did not decrease fluctuations of inhibitory synaptic sizes in that study [18].

How neural networks keep memories over months and years despite their synapses remodeling on (possibly much) shorter time scales is still an active research question. One hypothesis is that while synaptic configurations and the identities of receptive neurons drift slowly, connected areas adapt and follow the changes, thus maintaining network function [112, 113]. Another argument is redundancy [19]: there may be a large subset of synaptic configurations that implement equivalent network functionality. Synaptic connectivity could then change in 'irrelevant' directions without affecting network function. If the systematic synaptic changes concentrate on few 'relevant directions', whereas random synaptic fluctuations spread also over many irrelevant ones, then the signal-to-noise ratio in relevant directions may be much higher than measured in single synapses. This redundancy, related to the observation of low-dimensional activity (see Sec. 2.3.3), is a main argument used in the analysis of WP and NP, publication 1 [1], Chapter 3.

2.4.2 Reward-modulated plasticity

Long-term potentiation (LTP) or depression (LTD) describe lasting increases or decreases in synaptic strength, respectively. LTP (and similarly LTD) consists of an early and a late phase. Early LTP describes the initial, protein-independent strengthening of a synapse, which decays over a span of hours [21]. Late LTP describes the protein-dependent consolidation of the potentiation. The synaptic tagging and capture hypothesis [20–22] states that this synaptic consolidation requires the simultaneous presence of plasticity-related products/proteins (PRPs) and a synapse-specific 'tagged' state. Experience and the

concomitant synapse-local activity histories can induce early LTP and, potentially independently of it, a synaptic tag. This tag lasts for around 90 min, generating a time window for consolidation [21]. The production of the PRPs necessary for late LTP, which happens mainly at the soma, can be triggered also by plasticity events at other synapses and dendrites.

The creation of PRPs and hence the induction of LTP and LTD is influenced by the neuromodulator dopamine [22–24]. Dopamine is in large part mediated through volume transmission, i.e., it non-synaptically diffuses through the extracellular medium [26]. Because dopaminergic neurons also often project broadly to many areas, their modulatory effects are rather global [27], although they can also dynamically change in space and time [26]. As dopaminergic neurons encode reward- and general prediction errors [25], dopamine signals seem ideally suited to align local synaptic plasticity with network-level objectives. In three-factor learning rules, weight changes depend on a modulatory signal in addition to the pre- and postsynaptic activity histories [114]. The relationship between neuromodulation and plasticity (for example spike-timing-dependent plasticity (STDP)) can thereby be complex and differ between brain regions [114]. However, it is also often modeled as a multiplicative modulation [114, 115] by the reward R measured w.r.t. some baseline $\bar{R}$ [25],

$$\Delta w_{ij} = (R - \bar{R}) \cdot e_{ij}(\text{pre}, \text{post}), \tag{2.22}$$

where e_{ij} is an eligibility trace that temporally integrates contributions from pre- and postsynaptic activity. This integration allows connecting millisecond-resolution co-firing events of the pre- and postsynaptic neurons to reinforcing modulation delayed by seconds [116].

2.4.3 Weight (WP) and Node Perturbation (NP)

There are three learning paradigms: supervised learning, reinforcement learning (RL), and unsupervised learning. Supervised learning settings provide explicit output targets and detailed, vector-valued error feedback. Backpropagation [106] and biologically plausible variants [35] translate this feedback into synaptic changes that quickly reduce output errors. Unsupervised learning, on the other hand, operates without consideration of network output. Instead, the activity patterns in response to external input are shaped to build potentially useful representations. In an RL setting, the network output is evaluated, but the network does not have access to an output target. Instead, it receives only scalar reward/error feedback about its performance. In particular, the feedback contains only information about how good the network performs, but not about how it should change in order to improve. In addition, the feedback can often be temporally sparse and delayed. Many real world tasks lack a clear target and can be formulated as RL tasks.

Weight perturbation (WP) [12, 13] and node perturbation (NP) [14, 15] learning are two forms of reinforcement learning. They estimate the weight gradient by correlating random perturbations, of the weights or the summed weighted inputs of the nodes, with changes in task performance. If a weight perturbation lowers the task error (or equivalently increases reward), the weights are changed in the direction of the perturbation. If it increases the error, they change in the opposite direction. For sufficiently small perturbations and updates, and a smooth error function, the error changes approximately linearly with the weights so that an update lowers it in either case. NP applies random perturbations to the total inputs of the neurons. For temporally extended tasks, these are time-dependent. For beneficial perturbations, the weights are changed such that the neural dynamics change in the direction of the perturbation, and oppositely for unfavorable perturbations. Although individual updates have a (large)

random component, both learning rules yield an unbiased estimate for the true gradient [1, 13, 15].

The perturbed (pre-)activation z_{it}^{pert} of node i at time t in the presence of weight perturbations ξ_{ij}^{WP} or node perturbations ξ_{jt}^{NP} is (here and below in the notation of publication 1 [1], Chapter 3)

$$z_{it}^{\text{pert}} \overset{\text{WP}}{=} \sum_{j=1}^{N}(w_{ij} + \xi_{ij}^{\text{WP}})r_{jt}, \qquad z_{it}^{\text{pert}} \overset{\text{NP}}{=} \sum_{j=1}^{N} w_{ij}r_{jt} + \xi_{it}^{\text{NP}}. \tag{2.23}$$

Here r_{jt} is the jth input at time t, and the perturbations are iid. Gaussian-distributed with variance σ_{WP}^2 or σ_{NP}^2, respectively. In the presence or absence of perturbations the network obtains as error feedback the perturbed or unperturbed error, E^{pert} or E, respectively. The weight updates of WP and NP are [14]

$$\Delta w_{ij}^{\text{WP}} = -\frac{\eta}{\sigma_{\text{WP}}^2}(E^{\text{pert}} - E)\xi_{ij}^{\text{WP}}, \qquad \Delta w_{ij}^{\text{NP}} = -\frac{\eta}{\sigma_{\text{NP}}^2}(E^{\text{pert}} - E)\sum_{t=1}^{T}\xi_{it}^{\text{NP}}r_{jt}. \tag{2.24}$$

Here η is the learning rate, which is scaled by the variance $\sigma_{\text{WP|NP}}^2$ of the perturbations.

NP assumes that the total input $z_i = \sum_j w_{ij}r_j$ is a linear sum of inputs weighted by w_{ij}. The first part of the NP update, $-\frac{\eta}{\sigma_{\text{NP}}^2}(E^{\text{pert}} - E)\xi_{it}^{\text{NP}}$, is the stochastic estimate of the error gradient w.r.t. the total input z_{it}, i.e., $\mathrm{d}E/\mathrm{d}z_{it}$. The factor r_{jt} in the NP update then results from the application of the chain rule, in other words a backpropagation step: $\mathrm{d}E/\mathrm{d}w_{ij} = \sum_t(\mathrm{d}E/\mathrm{d}z_{it})(\mathrm{d}z_{it}/\mathrm{d}w_{ij}) = \sum_t(\mathrm{d}E/\mathrm{d}z_{it})r_{jt}$. Thus, while WP does not assume anything about how the weights (parameters) influence the performance of the network, NP does utilize some knowledge about the network.

WP and NP are versatile biologically plausible plasticity rules, as they impose little requirements on the implementing network. In particular, they require only a scalar reward (or error) signal that can be sparse and delayed. Further, WP and variants of NP [28] depend only on quantities that are, in principle, available at each synapse [28].

2.4.4 REINFORCE algorithms

The REINFORCE framework [29] (an acronym for 'REward Increment = Nonnegative Factor × Offset Reinforcement × Characteristic Eligibility', see Eq. (2.25)) describes a class of RL algorithms that learn input-output mappings in networks of stochastic units. They do so by correlating an 'offset reinforcement', i.e., a modulatory reward signal measured against some baseline, with a 'characteristic eligibility' that depends on pre- and postsynaptic activities. In this way, they are similar to three factor rules [114]. Critically, the activity of a node i, z_i^{pert}, (this section uses the notation from publication 1 [1]) is a stochastic function of its input weights w_{ij} and inputs r_j, with probability density function $f(z_i^{\text{pert}}; \mathbf{w}, \mathbf{r})$. The REINFORCE update rule reads [29]

$$\Delta w_{ij} = -\eta(E^{\text{pert}} - E)\cdot e_{ij}, \qquad e_{ij} = \frac{\partial \ln\left(f(z_i^{\text{pert}}; \mathbf{w}, \mathbf{r})\right)}{\partial w_{ij}}. \tag{2.25}$$

It is presented here with optimal baseline E and homogeneous learning rates $\eta_{ij} = \eta > 0$, while the original formulation includes also algorithms with inhomogeneities in these parameters. Note that the update depends on the concretely realized activations z_i^{pert} of the stochastic units.

For temporal tasks consisting of T time steps with error feedback provided only at the end of a trial, *episodic* REINFORCE update rules sum the contributions of Eq. (2.25) applied to each time step,

$$\Delta w_{ij} = -\eta(E^{\text{pert}} - E) \cdot \sum_{t=1}^{T} e_{ij}^t, \qquad e_{ij}^t = \frac{\partial \ln\left(f(z_{it}^{\text{pert}}; \mathbf{w}, \mathbf{r}_t)\right)}{\partial w_{ij}}. \tag{2.26}$$

Here e_{ij}^t is the characteristic eligibility at time step t and e_{ij} the eligibility *trace* that integrates update contributions over time.

How do REINFORCE algorithms work? Consider the non-episodic case $t = 1$. The characteristic eligibility e_{ij} involves the derivative of (the logarithm of) the probability density $f(z_i^{\text{pert}}; \mathbf{w}, \mathbf{r})$ w.r.t. the weight w_{ij}. A positive derivative means that increasing w_{ij} increases the chance that, given the same inputs r_j, node i reproduces the activity z_i^{pert} that was associated with the error E^{pert}. By multiplying with $-(E^{\text{pert}} - E)$, updates thus modify the distribution of node activities z_i^{pert} such that values associated with lower-than-expected error feedback become more probable, and vice-versa for higher errors.

Williams [29] proofs that, for homogeneous learning rates, the expected weight update $\langle \Delta \mathbf{w} \rangle$ is parallel to the weight gradient of the expected error $\langle E^{\text{pert}} \rangle$,

$$\langle \Delta \mathbf{w} \rangle = -\eta \nabla_w \langle E^{\text{pert}} \rangle . \tag{2.27}$$

Here the average is taken over the input distributions, the stochastic node activations (given weights and inputs) and the reward feedback (given input and node activations).

For a simplified sketch of the proof [29], consider non-episodic REINFORCE with a single stochastic node z_i^{pert}, $i = 1$, repeated input $\mathbf{r}$ and a deterministic error function. The only stochasticity is thus in the node activation z_i^{pert}. Then, by averaging over its realizations, we obtain

$$\begin{aligned}
\langle \Delta w_{ij} \rangle &= \int \mathrm{d}z_i^{\text{pert}} \underbrace{\left[-\eta \big(E^{\text{pert}}(z_i^{\text{pert}}) - E\big) \cdot \frac{1}{f(z_i^{\text{pert}}; \mathbf{w}, \mathbf{r})} \frac{\partial f(z_i^{\text{pert}}; \mathbf{w}, \mathbf{r})}{\partial w_{ij}} \right]}_{= \Delta w_{ij} \text{ given } z_i^{\text{pert}}} \cdot f(z_i^{\text{pert}}; \mathbf{w}, \mathbf{r}) \\
&= -\eta \int \mathrm{d}z_i^{\text{pert}} \big(\underbrace{E^{\text{pert}}(z_i^{\text{pert}})}_{\text{indep. of } w_{ij}} - E \big) \cdot \frac{\partial f(z_i^{\text{pert}}; \mathbf{w}, \mathbf{r})}{\partial w_{ij}} \\
&= -\eta \frac{\partial}{\partial w_{ij}} \int \mathrm{d}z_i^{\text{pert}} \, E^{\text{pert}}(z_i^{\text{pert}}) \cdot f(z_i^{\text{pert}}; \mathbf{w}, \mathbf{r}) + \eta E \frac{\partial}{\partial w_{ij}} \underbrace{\int \mathrm{d}z_i^{\text{pert}} \, f(z_i^{\text{pert}}; \mathbf{w}, \mathbf{r})}_{=1} \\
&= -\eta \frac{\partial \langle E^{\text{pert}} \rangle}{\partial w_{ij}}.
\end{aligned} \tag{2.28}$$

Here the third equality uses that E^{pert}, given z_i^{pert}, is independent of w_{ij}, and that $f(z_i^{\text{pert}})$, as a probability density function, is normalized to 1. In the first line, the probability density of z_i^{pert}, $f(z_i^{\text{pert}})$, cancels with the factor $1/f(z_i^{\text{pert}})$ from the definition of the characteristic eligibility, which motivates the use of the natural logarithm in its definition $e_{ij} = \partial \ln(f)/\partial w_{ij} = (1/f) \cdot \partial f / \partial w_{ij}$.

The episodic REINFORCE update rule, Eq. (2.26), looks reminiscent of the NP rule, Eq. (2.24).

Indeed, it follows from applying the REINFORCE framework to node (pre-) activations that are perturbed by iid. centered Gaussian white noise with variance σ^2_{NP} [14] when these are the linear sums of the inputs weighted by the respective weights: defining $z_{it} = \sum_{j=1}^{N} w_{ij} r_{jt}$ as the unperturbed node activation, and $\xi^{\mathrm{NP}}_{it} = z^{\mathrm{pert}}_{it} - z_{it}$ as its perturbation, its probability density function $f(z^{\mathrm{pert}}_{it}; \mathbf{w}, \mathbf{r}) = f_{\mathrm{gauss}}(z^{\mathrm{pert}}_{it}; \mu = z_{it}, \sigma^2 = \sigma^2_{\mathrm{NP}})$ is a Gaussian function. The characteristic eligibilities are thus (omitting the dependencies on $\mathbf{w}$ and $\mathbf{r}$)

$$
\begin{aligned}
e^t_{ij} &= \frac{1}{f(z^{\mathrm{pert}}_{it})} \frac{\partial f(z^{\mathrm{pert}}_{it})}{\partial w_{ij}} \\
&= \frac{\xi^{\mathrm{NP}}_{it}}{\sigma^2_{\mathrm{NP}}} r_{jt},
\end{aligned}
\qquad\qquad
\begin{aligned}
\frac{\partial f(z^{\mathrm{pert}}_{it})}{\partial w_{ij}} &= \frac{\partial f_{\mathrm{gauss}}(z^{\mathrm{pert}}_{it})}{\partial z^{\mathrm{pert}}_{it}} \cdot \frac{\partial z^{\mathrm{pert}}_{it}}{\partial w_{ij}} \\
&= \frac{\xi^{\mathrm{NP}}_{it}}{\sigma^2_{\mathrm{NP}}} f_{\mathrm{gauss}}(z^{\mathrm{pert}}_{it}) \cdot r_{jt}.
\end{aligned}
\tag{2.29}
$$

Inserting e^t_{ij} into Eq. (2.26), we obtain the NP rule, Eq. (2.24).

Appendix C shows how the REINFORCE framework is for temporally-extended tasks not directly applicable to WP, because it (implicitly) assumes that the (induced) perturbations of node activations are statistically independent. Weight perturbations, however, induce correlated node perturbations. We show that application of the REINFORCE framework to weight perturbations yields Hybrid Perturbation (HP)[1], and that WP can be recovered if the resultant updates are decorrelated.

2.4.5 Learning dynamics of WP and NP

Werfel et al. [14] formulated the modern version of the NP rule that uses simultaneous perturbations of all network nodes, using the REINFORCE framework [29]. Importantly, they advanced the understanding of WP and NP by comparing their learning dynamics with each other and with gradient descent (GD). They studied linear perceptrons that map N inputs r_j to M outputs z_i. The mapping $z_i = \sum_{j=1}^{N} w_{ij} r_j$ is described by the weights w_{ij}. The perceptrons learn associations between random input patterns r_j and target outputs $z^*_i = \sum_{j=1}^{N} w^*_{ij} r_j$ defined through a teacher perceptron with teacher weights w^*. On each trial, a random input pattern is drawn from an isotropic multivariate Gaussian distribution. The individual inputs r_j are iid. distributed with variance α^2/N; the total variance of the input pattern is α^2. In the perturbed trials, WP adds a perturbation ξ^{WP}_{ij} to the weights; NP perturbs the outputs by ξ^{NP}_i. The objective is to minimize the squared error, which for the perturbed trials reads

$$
E^{\mathrm{pert}} \overset{\mathrm{WP}}{=} \frac{1}{2} \left\| (\mathbf{W} + \xi^{\mathrm{WP}}) \mathbf{r} \right\|^2, \qquad\qquad E^{\mathrm{pert}} \overset{\mathrm{NP}}{=} \frac{1}{2} \left\| \mathbf{W}\mathbf{r} + \xi^{\mathrm{NP}} \right\|^2, \tag{2.30}
$$

where $\mathbf{W} = \mathbf{w} - \mathbf{w}^*$ is the weight mismatch.

By averaging over perturbations and input patterns, ref. [14] calculate how the expected error evolves with the number of updates. They find that the expected error exponentially approaches a final, residual error E_f,

$$
\langle E \rangle (n) = \left(\langle E \rangle (0) - E_f \right) a^n + E_f. \tag{2.31}
$$

In the absence of noise in the learning rule and for infinitesimal perturbations, GD, NP and WP lower the expected error at the same rate, which is $\lambda^l = -\ln(a) \approx 1 - a$ with $a = 1 - 2\eta$ [14], where η is the

learning rate. It turns out, however, that in the studied setting GD can operate at a higher learning rate than NP, which in turn allows a higher rate than WP. Thus, when compared at the learning rate η^* that leads to the fastest convergence of the expected error, GD learns faster than NP, which learns faster than WP [14],

$$a_{\mathrm{GD}} = 1 - \frac{1}{N+2}, \qquad E_f^{\mathrm{GD}} = 0, \tag{2.32}$$

$$a_{\mathrm{NP}} = 1 - \frac{1}{(M+2)(N+2)}, \qquad E_f^{\mathrm{NP}} = \tfrac{1}{8}\sigma_{\mathrm{eff}}^2 MN\frac{M+4}{N+2} \approx \tfrac{1}{8}\sigma_{\mathrm{eff}}^2 M^2, \tag{2.33}$$

$$a_{\mathrm{NP}} = 1 - \frac{1}{(MN+2)(N+2)}, \qquad E_f^{\mathrm{NP}} = \tfrac{1}{8}\sigma_{\mathrm{eff}}^2\left(M^2N\frac{M+2}{MN+2} + 12\frac{MN+2}{N}\right) \tag{2.34}$$

$$\approx \tfrac{1}{8}\sigma_{\mathrm{eff}}^2 M^2. \tag{2.35}$$

Here the variance of an input component is, without loss of generality, set to 1, and σ_{eff} is the effective variance of the activity perturbation that the weight or node perturbations cause. Concretely, $\sigma_{\mathrm{NP}}^2 = \sigma_{\mathrm{eff}}^2$ and $\sigma_{\mathrm{WP}}^2 = \sigma_{\mathrm{eff}}^2/N$ for a fair comparison. The convergence rates are thus $\lambda_{\mathrm{GD}}^l \approx 1/N$, $\lambda_{\mathrm{NP}}^l \approx 1/MN$ and $\lambda_{\mathrm{WP}}^l \approx 1/MN^2$. In the studied task, GD thus learns faster than NP by a factor of M, which in turn learns faster than WP by a factor N [14].

If GD used the gradient averaged over the full input distribution, it could achieve zero error in a single update. However, due to the randomly drawn input patterns that lie in an N-dimensional space, GD can, in a single trial, only update the weights 'from a single input direction'. This can be seen from Eq. (2.30), where the error in a single trial only depends on the projection of the weight mismatch $\mathbf{W}$ on the current input pattern $\mathbf{r}$. Consequently, GD has to average over N trials to obtain an estimate of the full gradient. This lowers the optimal learning and convergence rate by the same factor N, an effect that equally holds for WP and NP. Phrased differently, the difference between the gradients on a single trial and over the full input distribution represents 'gradient noise', which necessitates averaging over multiple trials [14].

WP and NP do not have access to the true gradient (of a single trial), but estimate it by applying random perturbations to the MN weights or M nodes, respectively. Because the perturbations are isotropic, only a fraction $1/MN$ or $1/M$ of their variance, respectively, is aligned with the true weight or node gradient. However, also the part of a perturbation that is perpendicular to the gradient contributes to the updates. As it is approximately $\sqrt{MN}$ or $\sqrt{M}$ times larger than the gradient-parallel component, it requires MN or M updates, respectively, to average it out. This explains why the convergence rates of NP and WP, compared to GD, scale inversely with the dimension of the perturbations, which is smaller for NP [14].

In their abstract, ref. [14] acknowledge that '... these statements [the scaling of the convergence rates] depend on the specifics of the learning problem, such as the input distribution and the target function, and are not universally applicable'. Indeed, Chapter 3 and publication 1 [1] analyze WP and NP in a setting where (throughout most of the article) the input is not random but the same in each trial and has an additional temporal dimension. This, together with inputs that are effectively low-dimensional (cf. Sec. 2.3.3), allows WP to perform similarly or better than NP, challenging the generality of the intuitions from the previous paragraph.

2.4.6 Plausible node perturbation learning

For the update of the NP rule, a synapse from neuron j to neuron i has to calculate an eligibility trace $e_{ij} = \sum_t \xi_{it} r_{jt}$ that involves the perturbation ξ_{it} of the postsynaptic input [12, 13, 28]. The postsynaptic

membrane potential or input current may be accessible to the synapse – but how can it distinguish perturbatory from normal inputs? One way is to assume that perturbatory fluctuations have a faster time course than the normal neuronal activity, so that the fluctuations of the postsynaptic total input x_{it} can be approximated as $\xi_{it} \approx x_{it} - \bar{x}_{it}$, where $\bar{x}_{it}$ is low-pass filtered [28, 117]. Each relatively sharp perturbatory fluctuation then adds a short, larger-amplitude excursion to $x_{it} - \bar{x}_{it}$, the 'perturbation part', with a trailing longer, lower-amplitude excursion of opposite sign, the 'relaxation part' [28]. These terms cancel: for continuous inputs that assume a constant value as $t \to \pm\infty$, the temporal integral over x_{it} equals that of $\bar{x}_{it}$, so that their difference has zero integral. If the presynaptic rate r_{jt} changes only slowly compared to the time constant of the low-pass filtering, then the contribution from the relaxation part to the eligibility trace negates that from the perturbation part. The 'relaxation contribution' is further undesirable as it matches perturbations with temporally unrelated (more distant and acausal) inputs in the eligibility trace. NP based on this eligibility trace does hence not work with delayed reward [28].

Ref. [28] suggests that a supralinear transformation $S(x_{it} - \bar{x}_{it})$ of the recent total input change can emphasize its larger perturbation part while de-emphasizing its smaller relaxation part. Correspondingly, the modified NP rule of [28] reads

$$\Delta w_{ij} = -\eta \left(E - \bar{E}\right) \sum_{t=1}^{T} S\left((x_{it} - \bar{x}_{it}) r_{jt}\right), \tag{2.36}$$

with $S(x_{it} - \bar{x}_{it})$ replacing the actual perturbation ξ_{it}^{NP} as used in vanilla NP, and E and $\bar{E}$ being the current error (or negative reward) and the expected error, respectively. The difference $-(E - \bar{E})$ is thus a reward prediction error. In practice, ref. [28] uses a running exponential average of the error to estimate the expected error $\bar{E}$. The nonlinearity $S(x) = x^3$ is chosen as a cubic function, which is shown to lead to warped but reasonable gradient estimates [28]. The used perturbations are given by iid. random additions Δx_{it} to the pre-activations x_{it} at sparse, discrete time points. The sparseness of the perturbations together with their sharpness, as opposed to the otherwise smooth evolution of neuronal activity, seems to be ideal to separate the perturbation part from the relaxation part in the learning rule.

Result section M of publication 1 [1] compares the NP variant of ref. [28] with vanilla WP and NP, with the unperturbed reward replaced by a running average of recent rewards, for the delayed non-match-to-sample task from the same article. This work was done by Christian Klos. It demonstrates successful and comparable WP and NP learning of recurrent connectivity with the more realistic use of a predicted instead of an unperturbed reward.

2.5 Further mathematical tools

2.5.1 Isserli's theorem

Let $\mathbf{X} \sim \mathcal{N}(\mu, \mathbf{\Sigma})$ be an n-dimensional random variable that follows a multivariate Gaussian distribution with mean μ and nonsingular covariance matrix $\mathbf{\Sigma}$. The probability density function $f(\mathbf{X} = \mathbf{x})$ for observing $\mathbf{x}$ is explicitly given by

$$f(\mathbf{X} = \mathbf{x}; \mu, \mathbf{\Sigma}) = \frac{1}{\sqrt{(2\pi)^n \det \mathbf{\Sigma}}} \exp\left(-(\mathbf{x} - \mu)^T \mathbf{\Sigma}^{-1} (\mathbf{x} - \mu)\right). \tag{2.37}$$

For centered distributions, i.e., $\mu = 0$, Isserli's theorem [118] states that the moments of X,

$$\langle X_{i_1} X_{i_2} \cdots X_{i_k} \rangle = \int \mathrm{d}x_{i_1} \cdots \int \mathrm{d}x_{i_k} \; (x_{i_1} x_{i_2} \cdots x_{i_k}) \; f(\mathbf{X} = \mathbf{x}; 0, \mathbf{\Sigma}), \tag{2.38}$$

are given by

$$\langle X_{i_1} X_{i_2} \cdots X_{i_k} \rangle = \sum_{p \in P_k^2} \prod_{\{m,l\} \in p} \langle x_{p_m} x_{p_l} \rangle \, . \tag{2.39}$$

Here P_k^2 is the set of partitioning the indices $i_1, i_2, \cdots, i_k$ into pairs, and the product is over all pairs in a given partition. As an example, consider the 4th order moment $\langle X_1 X_2 X_3 X_4 \rangle$, for which $P_4^2 = \Big\{ \big\{ \{1,2\}, \{3,4\} \big\}, \big\{ \{1,3\}, \{2,4\} \big\}, \big\{ \{1,4\}, \{2,3\} \big\} \Big\}$. Consequently,

$$\langle X_1 X_2 X_3 X_4 \rangle = \langle X_1 X_2 \rangle \langle X_3 X_4 \rangle + \langle X_1 X_3 \rangle \langle X_2 X_4 \rangle + \langle X_1 X_4 \rangle \langle X_2 X_3 \rangle \, . \tag{2.40}$$

Moments of odd number are zero, consistent with the inability to partition the indices into pairs.

2.5.2 Lambert W function and solutions

The Lambert W function is the multivalued inverse of the function $w \to w\mathrm{e}^w$ [119],

$$w = W_k(z) \qquad \Rightarrow \qquad w\mathrm{e}^w = z, \tag{2.41}$$

where the index k denotes its kth branch. W is also called the 'product logarithm', highlighting its broad similarity to the complex logarithm, which inverts the function $w \to \mathrm{e}^w$. Like the complex logarithm, it has different branches, as shown in Fig. 2.1a). The 0th and -1st branch are the only ones that map (part of) the real axis in the z-plane to real values in the w-plane [119], Fig. 2.1a). The two solutions coincide at the branch point $z = (-1)\mathrm{e}^{(-1)}$ where they yield $W_{0|-1}(-\mathrm{e}^{-1}) = -1$, which can be easily checked to satisfy Eq. (2.41). For $z > 0$ there is only one real solution (on the principal $k = 0$ branch), for $-\mathrm{e}^{-1} \leq z < 0$ there are two, and for $z < -\mathrm{e}^{-1}$ there is no real solution [119]. In the analysis of network dynamics in publication 2 [2] this will be connected to the emergence of oscillating solutions.

Comparing the solutions of the 0th and -1st branch with those of the other branches (for $z < 0$, markers in Fig. 2.1a)) shows that the other solutions have smaller real part and (in amplitude) markedly larger imaginary parts. In the context of publication 2 [2], Chapter 4, where W_k appears in the computation of the complex frequency of an exponential decay, these large imaginary parts characterize solutions with unrealistically large oscillation frequencies that decay quickly.

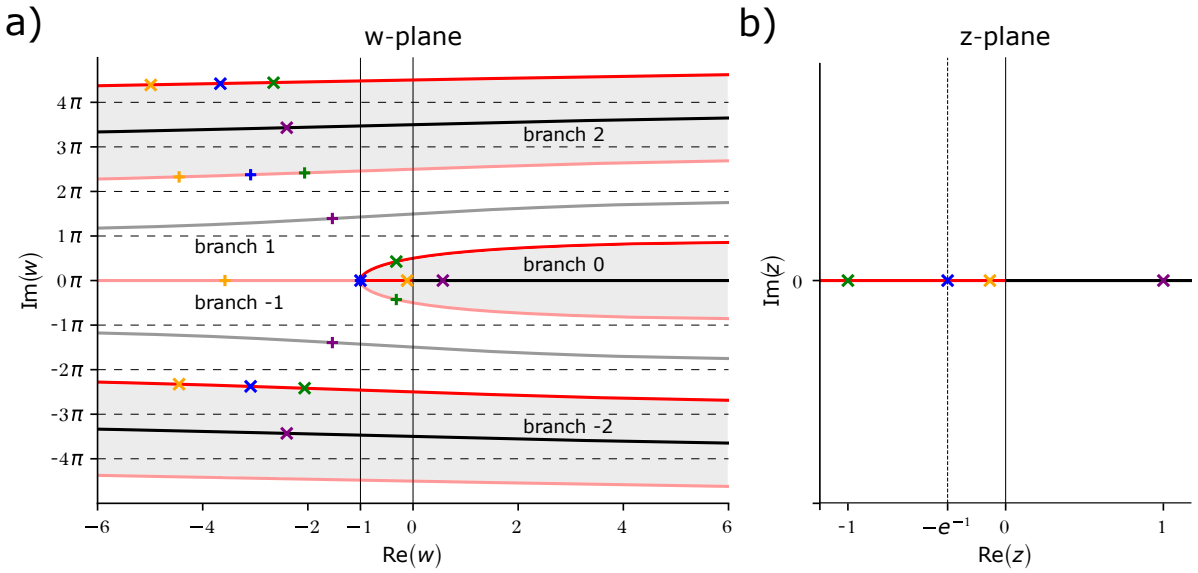

Figure 2.1: Images $\{w = W_k(z) \ : \ z \in \mathbb{C}\}$ of the first branches $k = -2, \cdots, 2$ of the Lambert W function. **a)** w-plane: Red lines show the images of the negative real line $z < 0$, black lines the images of the positive real line $z > 0$. The values $W_k(z)$ for $z = -1, -e^{-1}, -0.1, 1$ are marked in green, blue, orange and purple, respectively (compare with b)). Even branches use thick lines, crosses as markers and have images highlighted by a gray background. Odd branches use faint lines, pluses as markers and white backgrounds. Red lines mark the branch cuts and belong to the respective lower branch. Note that $w \to we^w$ maps all w values marked in the same color to the same z values as shown in b). Note also that only the 0th and -1st branch contain (part of) the real axis in their image. **b)** z-plane: The red and black line highlight the negative / positive real numbers, which are the pre-images of the respective lines in a).

CHAPTER 3

Analysis of weight and node perturbation

This chapter consists of the following publication (publication 1) and a summary thereof:

[1] **P. Züge**, C. Klos and R.-M. Memmesheimer
Weight versus Node Perturbation Learning in Temporally Extended Tasks: Weight Perturbation Often Performs Similarly or Better
Physical Review X **13** (2023) 021006

The article is attached in full in App. A. Its supplementary material (supplement 1 [120]), available from the same DOI, is attached in full in App. B. In addition, App. C relates WP and hybrid perturbation (HP), developed in publication 1, to the REINFORCE framework [29].

Contribution statement The article was conceptualized by all authors. The theoretical analysis was performed by me. Further, I initiated the study of correlated inputs with low effective dimension and discovered the interference of node perturbations with unrealizable target components. I performed the numerical simulations that directly test the theoretical analysis, except for the simulated error dynamics for learning multiple subtasks. The results of these simulations underlie Figs 1-7 of publication 1 [1] and Figs S1-S4,S6-S9 of its supplement [120]. I wrote the majority of the results part, except the sections about the DNMS and MNIST tasks. I further contributed strongly to the planning and revision of the other parts of the article. I also wrote appendices A3 and A4, as well as the supplement [120].

3.1 Summary

This chapter gives a brief summary of the results from publication 1 [1]. For the details, consult publication 1 (App. A) or supplement 1 [120] (App. B) directly. Chapter 5 provides further discussion of the results and places them into a wider scientific context.

Animals can learn to perform RL tasks in the presence of sparse, delayed rewards (see Sec. 2.4.2). The underlying neural circuits thereby work in an inherently noisy fashion (Sec. 2.4.1). Two basic RL rules that can exploit noise in neural computations are WP and NP (Sec. 2.4.3). NP was believed to be highly superior to WP due to its massively smaller perturbation dimension [14] (Sec. 2.4.5): there are many more weights than nodes, so that the projection of a random perturbation on the relevant gradient is relatively smaller for WP. This intuition was substantiated by the theoretical analysis of linear networks learning to map random inputs in a student-teacher task: NP can in that case learn faster than WP by a factor of N, which is the number of inputs to a neuron [14] (Sec. 2.4.5). It is therefore, in contrast to WP, often considered as a benchmark [34–36] and as actually implemented in biological neural networks [28, 30, 31, 121].

The setting analyzed in ref. [14], which cemented NP's reputation as the superior learning rule, features unstructured inputs without temporal extent. Neuronal activity, however, is typically correlated (depending on the timescale and averaging procedure, see Sec. 2.2.1), as characterized by a relatively low effective dimension (Sec. 2.3.3). Also, tasks extend in time, and sparse, delayed rewards necessitate relating them to prior network activity (Sec. 2.4.2).

For this reason, we analyze WP and NP for time-extended tasks with correlated inputs. Concretely, we consider learning a single linear input-output mapping with effectively low-dimensional inputs. Later we extend this to learning multiple input-output mappings. We calculate the evolution of the expected error for a quadratic error function, which reveals the dependencies of the learning rules on different task properties. The convergence rate and asymptotic value of the expected error serve to compare the two learning rules.

We find that

> WP learns a single input-output mapping not slower but equally fast as NP. Further, WP can reach a lower final error. We generalize these findings by demonstrating that WP can perform similarly to or better than NP also for more complex machine learning and behavioral tasks.

Through mathematical analysis, we develop a clear mechanistic understanding of the learning dynamics. We define task-relevant and -irrelevant weights (more precisely: weight space directions), with irrelevant weights mediating zero inputs. We find in particular that

> The convergence speed of WP and NP is determined not by the number of weights or nodes, respectively, but by the number of task-relevant weights.

For correlated inputs, this number is generally between the number of nodes and the number of weights, making the task easier than expected for WP and harder for NP.

> The asymptotically achievable error of NP is proportional to the temporal dimension T, while that of WP is proportional to the effective input dimension N_{eff}. As this is always equal to or smaller than T, WP can achieve a lower final error.

We find differences in the weight and learning dynamics of WP and NP that might serve to distinguish them in neurobiological experiments: First,

> For WP but not NP, irrelevant weights perform a random walk.

We show that weight decay counteracts and stops this 'weight diffusion'. In the presence of input noise, irrelevant weights diffuse also for NP. Their diffusion, which is still stronger for WP, then contributes to the task error. It stops at a finite spread like in the case of weight decay, due to the error feedback. Despite the increased number of weights affecting the task error, and despite the stronger diffusion of irrelevant weights for WP, we find that

> WP can outperform NP also in the presence of input noise. The dynamics and final spreads of the irrelevant weights remain qualitatively different between the learning rules, allowing to distinguish them.

Second,

> Learning multiple subtasks slows down WP but not NP (in the absence of input noise).

This is because WP's random changes to weights that are irrelevant in one subtask can become relevant in another, where they interfere with learning. On the flip side,

> WP benefits from batch learning, while NP surprisingly does not.

In behavioral experiments, this would correspond to rewarding action sequences versus single actions. We further consider target outputs that contain components not present in the inputs. As a third distinguishing characteristic, we show how such

> Unrealizable target components interfere with node but not weight perturbations.

This is because unrealizable perturbations parallel to these components influence the error to linear order, contributing only reward noise.

Using the obtained insights,

> We develop two new learning rules: Hybrid perturbation (HP) and WP0.

HP combines weight perturbations with NP updates (see also App. C), producing relevant and realizable perturbations while using eligibility traces to solve part of the credit assignment problem. Like NP, HP does not update irrelevant weights; however, it preferentially updates weight directions that mediate strong input components, which introduces a bias. Accordingly, it works best when all latent inputs have the same or similar strength, or if it is sufficient to learn the weights mediating strong inputs. WP0 is a WP variant that does not update weights mediating zero inputs. It thus works best when the inputs within a trial are sparse. For favorable task properties, HP and WP0 can learn also multiple subtasks as fast as NP, while not suffering from unrealizable target components and reaching a final error as low as WP. As a new variant of NP, we also consider NP with temporally correlated perturbations and find that

> Temporally correlated perturbations improve NP when the inputs are similarly correlated.

It behaves then similar to NP on a task with reduced temporal dimension and can achieve final errors between those of WP and NP with uncorrelated perturbations.

For tasks with heterogeneous latent input strengths, we find that

> WP and NP learn weights connected to strong and weak inputs at different speeds

and demonstrate this in a reservoir computing task.

Going beyond the assumptions made in the theoretical analysis, we apply WP and NP to more complex tasks, concretely training an RNN on a delayed non-match-to-sample (DNMS) task using more biologically plausible reward baselines and eligibility traces. Furthermore, we train multi-layered networks of nonlinear neurons on an RL version of the MNIST (Modified National Institute of Standards and Technology database) task. We find that

> WP performs similarly to or better than NP also for more complex tasks and network architectures.

To conclude, we have shown that the performance of WP and NP depends critically on the task setting: For time-extended or batched tasks with correlated inputs where single trials capture most of the task's content, WP can perform better than NP. This suggests WP as a relevant benchmark and, together with its even simpler biological implementation, as a plausible candidate for learning in the brain. The differences in the weight and learning dynamics may allow to experimentally distinguish the learning rules.

CHAPTER 4

Cooperative coding

This chapter consists of the following pre-publication (publication 2, version 2) and a summary thereof:

[2] **P. Züge** and R.-M. Memmesheimer
Cooperative Coding of Continuous Variables in Networks with Sparsity Constraint
bioRxiv (2024):2024.05.13.593810v2

The article is attached in full in App. D.

Contribution statement The article was conceptualized by both authors. The theoretical and numerical analysis was jointly planned, and performed by me. I suggested studying also the encoding of higher-dimensional stimuli and the dynamical speedup through delayed, balanced inhibition. I wrote nearly all of the initial draft and the majority of the supporting information (S3-S7). Main text and supporting information were revised by both authors.

4.1 Summary

This chapter gives a brief summary of the results from publication 2 [2]. For the details, consult publication 2 (App. D) directly. Chapter 5 provides further discussion of the results and places them into a wider scientific context.

Neurons in biological neural networks encode continuous sensory and intrinsic variables in their joint activity [6]. Individual neurons thereby respond to a rather wide range of stimuli, and any sufficiently strong stimulus activates many neurons. The neuronal response profiles are thereby similar, in the sense that for any neuron there are typically many neurons with highly overlapping responses [41, 42]. In the cortex, (excitatory) synaptic connections between such highly similarly-tuned principal neurons are more likely and stronger [43–45]. Also their functional connectivity, which includes polysynaptic inhibitory contributions, is excitatory [46, 47].

One way to progress our understanding of biological neural networks is to discover normative explanations for experimental observations. The observed predominantly excitatory coupling between similarly-tuned principle neurons is in contrast to normative theories that optimize coding with a limited number of spikes and predict inhibitory like-to-like connectivity [48, 49, 82, 122] (Sec. 2.2.3). In addition, recurrent excitation can amplify noise [50] and increase response times [51, 52] (Sec. 2.3.2), which seems disadvantageous for fast stimulus processing. This poses the question:

> Why do similarly-tuned principal neurons predominantly excite each other?

To answer this question, we develop a novel cooperative coding scheme, in which neurons utilize recurrent connections to harness the computations already performed by similarly-tuned neurons. This cooperative sharing of computations lessens the need for individual feedforward processing. Viewed differently, neurons share their access to feedforward information specifically with other neurons that also need it. The dynamic interplay of feedforward input and recurrent interactions then forms the network response. As a result,

> Cooperatively coding networks can replace many feedforward and less specific recurrent connections by few specific recurrent ones, thereby reducing the number of required synapses.

Crucially, the resulting connectivity between neurons with very similar receptive fields (RFs) is net excitatory. Our results thus show that

> The observed excitation between similarly-tuned neurons can be a sign of a cooperative coding scheme that reduces the number of synapses.

This suggests the number of synapses and space constraints as an important factor in shaping biological neuronal networks.

Concretely we analyze linear rate networks with responses characterized by their neuronal RFs. We contrast our recurrent, cooperatively coding networks with purely feedforward networks that implement the same response, defined as their steady states.

> For the feedforward implementations, the number of required synapses per neuron increases linearly with the RF size n_{RF}. The cooperatively coding networks, in contrast, can in principle implement RFs of any width with a fixed number of synapses,

making n_{RF} a measure for the number of saved synapses. They do so by recurrently propagating activity to other neurons with progressively less overlapping RFs. This activity propagation causes an increase in response time that scales with the RF size. We thus find that

> Saving synapses comes at the cost of increased response times, with higher savings and slower responses for wide RFs.

We calculate the scaling of this trade-off analytically and find that a cooperatively coding network with purely excitatory synapses is rather slow: when encoding a continuous, one-dimensional stimulus, its response time scales quadratically with the RF size (and the number of saved synapses). However,

> The network response time can be massively reduced by introducing balancing inhibition that tracks excitation with a small delay [63, 64] (Sec. 2.3.4) and implements balanced amplification [51] (Sec. 2.3.5).

The mechanism underlying the speedup is as follows: Balancing inhibition allows (much) larger excitatory weights, as the steady-state inhibition cancels the additional excitation. During the response build-up, however, the delayed inhibition does not yet fully cancel the additional excitation, allowing strong interactions during this brief 'window of opportunity'. In contrast to the net interactions between neurons, this allows a propagation of activity *changes* through the network. For strong balance, this leads to oscillations and can destabilize the network. At a critical balance, which is at the onset of oscillations, network activity converges fastest to the steady state, and the scaling of the response time with RF size improves from quadratic to linear. Furthermore,

> Spike frequency adaptation (SFA), modeled as a low-pass filtered, inhibitory adaptation current [123], equally improves the scaling of the response time with the RF size.

Neurons are often selective to multiple stimulus attributes [124]. We therefore study the encoding of a two-dimensional stimulus with neurons with linear and nonlinear mixed selectivity [124] (in terms of the stimulus attributes). For linear mixed selectivity, the scaling of the response time with RF size is the same as for the one-dimensional stimulus, but improves by a constant factor. For nonlinear mixed selectivity, we find that

> When encoding a two- instead of one-dimensional stimulus, the scaling of the response time with RF size improves from quadratic to linear for excitatory networks. For networks with balancing inhibition or SFA it improves from linear to square-root-like.

Finally, we estimate the metabolic cost of synaptic transmission, measured by the L1-norm of transmitted currents. We find that the excitatory cooperatively coding networks use as much energy as the feedforward implementations. Adding balancing inhibition or SFA increases energy demand through the cancelation of currents. Balancing inhibition further requires additional synapses and neurons. The brain might thus invest some synapses, neurons, space, and energy in balancing inhibition to retain a reasonable response speed. Inhibitory neurons may also be required anyway for different reasons (Chapter 5).

In conclusion, our results indicate that recurrent like-to-like excitation might implement a novel cooperative coding scheme that minimizes the number of required synapses. This suggests the number of synapses and limited space as a crucial constraining factor in shaping neural networks. Finally, we show how balancing inhibition and a novel mechanism involving SFA can speed up neural computations.

CHAPTER 5

Discussion

This thesis studies biologically plausible neuronal plasticity and computation that are subject to physiological constraints. The constraints include the locality of learning rules, observed random fluctuations of network parameters and activity, as well as limited space and energy. Another constraint, which holds for many tasks, is that the optimal network output is unknown, but has to be inferred from sparse, delayed rewards. Making the learning still more difficult, synapse-local learning implies that individual synapses have to rely on their limited information to optimize network-level objectives.

We study the learning dynamics and their dependence on different task parameters for two such local RL rules, WP and NP. Their individual synaptic updates are highly stochastic and can even seem random, yet jointly they improve the network's ability to obtain reward. A similar theme of emergence is present on the level of neuronal activity, where we interpret local recurrent excitation between similarly-tuned cells, as observed in the cortex, as implementing a novel cooperative coding scheme. Cooperatively coding neurons thereby share computations with similarly-tuned neurons that also need them to establish a desired network response, thereby minimizing the number of required synapses. This allows us to explain the observed connectivity as an optimal solution given a constraint on synapse number.

Both our studies transform the view of seemingly disadvantageous properties of biological neural networks – spontaneous weight fluctuations or like-to-like excitation – and interpret them as parts of mechanisms – WP or cooperative coding – that optimize global objectives: maximizing reward with good performance or implementing a network response with few synapses.

Models and methodology We develop a mechanistic understanding, of the learning dynamics of WP and NP and of the activity dynamics of cooperatively coding networks, by studying abstract models that are as simple as possible while still capturing the relevant aspects under study. In particular, we mainly study linear networks of rate neurons (Secs. 2.3.1,2.3.2), which are amenable to exact analytical analysis. To understand them, we employ tools from linear algebra, statistics, and the theory of dynamical systems. Furthermore, dimensionality arguments play an important role in both studies. We confirm our results in numerical simulations, of linear and partially also of nonlinear rate networks with more complex architectures. In the tradition of normative approaches [11, 48, 53, 54, 125], we show how our models optimize an objective, i.e., the obtained reward or the number of synapses. The objective then provides a possible explanation for experimental observations, i.e., activity-independent plasticity or like-to-like excitation.

Expressivity of the used models Our linear rate models suffice to study the core problems investigated in this thesis: To understand the learning dynamics of WP and NP, we need to know how they solve the credit assignment problem, i.e., how they update each weight given the applied perturbations and obtained reward. Adding a neuronal nonlinearity to our single-layer linear networks would thereby not affect the critical linear input summation, but only effectively modify the dependence of the loss function on the network output. The developed concepts of task-relevant and -irrelevant weights, as well as of unrealizable target components, depend on the input structure and apply to such nonlinear networks in the same way. Further insights that build on these concepts, like the different dependencies of WP and NP on the number of subtasks, carry over to nonlinear networks, as demonstrated on the MNIST task.

For the analysis of cooperative coding, we ask how a given response can be constructed using different sets of recurrent and feedforward connections, without interest in a particular network response per se. The important aspect captured in our simple model is that a 'large part' of the input-output transformation of a neuron is already present in the output of neurons with highly similar RFs [41, 42, 44]. Further, these feature neurons have broad, strongly overlapping responses that provide indirect access to many input neurons. In contrast, input neurons have delta-like activity in the input space. These points still hold for cooperatively coding networks of nonlinear neurons, and presumably also for spiking neural networks that are well described by rate networks. In either case, the core insight is that a desired response can be constructed with fewer synapses from few relevant, recurrent inputs (and few feedforward inputs) than from feedforward inputs alone.

Scientific context of publication 1 (WP and NP learning) Basic research aims to improve scientific theories to better understand and predict natural phenomena. This thesis advances the understanding of two basic RL rules, WP [12, 13] and NP [14, 15] (Sec. 2.4.3), which are known for decades. Based on the intuitive argument that there are more weights than nodes to perturb and on theoretical analysis [14], NP was widely believed to be far superior to WP. NP is therefore, and because of its wide applicability as a black-box optimization algorithm, often used as a benchmark comparison for other biologically plausible learning rules [34–36]. It is also more often than WP considered as actually implemented in the brain [15, 28, 30, 31, 121]. Our findings offer a more nuanced evaluation, demonstrating that WP is actually preferred for many realistic conditions. It is especially competitive for long but relatively low-dimensional tasks, and when single trials provide already a lot of information about the full task or can be combined into batches. A concrete example is song learning in certain birds, where a single, stereotyped input sequence with high temporal precision drives song production [32].

The REINFORCE framework [29] describes a family of perturbation-based RL rules that increase the probability of reproducing stochastic activity that led to higher rewards. Applying this framework to perturbations of the summed neuronal inputs yields NP [14]. The weight perturbations of WP induce time-correlated node perturbations. The REINFORCE framework, however, assumes that (induced) neuronal perturbations are statistically independent across time and neurons. For temporally-extended tasks, it is thus not applicable to perturbations of the weights. Appendix C shows that disregarding this violation of assumptions and applying it anyway results in hybrid perturbation (HP) updates, which we introduce in publication 1 [1]. For specific task settings, HP combines the advantages of WP and NP, at the cost of biased updates. Decorrelating the updates of HP results in WP updates [1].

In deep and recurrent neural networks, applied activity perturbations induce further activity perturbations in other neurons. To account for the mismatch between applied and actual perturbations, ref. [126] propose to use the actually induced perturbations in the update rule, which they term activity-based

node perturbation (ANP). While ANP did not yield noticeable improvements over NP for the studied deep neural networks, it can do so in recurrent neural networks with online reward feedback [127]. In our single-layer networks, however, applied and actual node perturbations agree. Refs. [126, 127] also decorrelated neuronal activity, either between layers or continuously using a decorrelation matrix that co-evolved with training [128], which improved ANP. Interestingly, in ref. [127] WP and NP did not profit from such decorrelation. We have speculated that HP should profit greatly from such decorrelation, as it removes its bias. WP0 might benefit, too, if a partial decorrelation would project the weakest input components to zero, allowing it to prevent many updates of irrelevant weights that can be disadvantageous when learning multiple subtasks.

Our numerical experiments included training a layered network of nonlinear units on MNIST. By comparing to ablated shallow and linear networks, we discovered that NP learning was, compared to WP learning, more strongly impaired by nonlinear activation functions and the layered architecture. Since then, ref. [129] have studied NP learning in linear feedforward networks with a hidden layer analytically. Using a mean field approach, they found that NP learning can become unstable, triggered by diffusive growth of the hidden layer weights. This happens in the presence of target noise, consistent with our work.

Scientific context of publication 2 (cooperative coding) One goal of theoretical neuroscience is to find normative explanations for experimentally observed properties of biological neural networks. Such theories can for example predict the ratios of excitatory to inhibitory neurons [11], of the volumes taken up by neuronal wiring to neuronal cell bodies [53] or the distribution of axon diameters [54] from optimizing for limited space. Other experimental findings that optimize space usage include cortical folding [10] which increases cortical surface area, and, to some extent [130, 131], the wiring-length-minimizing arrangement of brain areas [132]. In the case of the observed functional and physical excitation between highly similarly-tuned cortical neurons [43–47], such a normative reason is still to be found. To the contrary, there are several theoretical arguments against such connectivity, favoring like-to-like inhibition instead [48–52, 82, 122]. An exception is ref [50], which proposes that although recurrent excitation and the concomitant neuronal correlations limit the capacity to encode information, they increase 'consistency' across neurons and time, improving the ability of suboptimal readouts to decode information. In this spirit, ref [133] had developed a simple readout from an intermediate layer of recurrently-connected and correlated neurons to solve classification tasks. Their 'recurrent readout' thereby works with a small, finite number of synapses per neuron, but requires more neurons and total synapses than a simple fully-connected readout.

Cooperative coding provides an objective that, when optimized, potentially explains the found recurrent excitation: the number of synapses required to implement a desired response. Intuitively, if the same effect can be obtained with fewer synapses, this could offer benefits in terms of reduced demand for space and energy. However, cooperative coding did not reduce metabolic costs, which we approximated by the L1-norm of synaptic currents. It is possible that a more detailed metabolic model that takes into account additional costs for synapses that are more distant from the soma [134], such as feedforward versus recurrent synapses, could result in slight metabolic savings. We found that networks with EI-balance or SFA had higher metabolic costs than a feedforward implementation, besides requiring additional neurons. Inhibition might, however, be required for reasons unrelated to cooperative coding, such as stabilizing activity [58, 59] and plasticity [100], causing asynchronous irregular spiking [83], expanding neuronal dynamic ranges [135], and implementing active computations [100, 136, 137].

Impact Our analysis of WP and NP reveals that WP performs for many relevant tasks much better than expected, and often better than NP, correcting the prior evaluation of NP as strictly superior. This suggests WP as a similarly relevant benchmark learning rule, and, in combination with its even simpler implementation, as a plausible candidate for learning in the brain. WP's demonstrated suitability for biologically plausible RL transforms the interpretation of experimentally found activity-independent plasticity (Sec. 2.4.1), suggesting that part of it may represent weight fluctuations that perform a useful exploration of the parameter space, thereby enabling targeted plasticity. This targeted plasticity would be another part of activity-independent plasticity, as WP updates depend only implicitly on activity through the obtained rewards. Finally, the developed insights into WP and NP learning, alongside the potential improvements they enable, including WP0 and HP, are relevant to model-free RL methods in the context of machine learning. WP and NP can have advantages over RL schemes that rely on backpropagating errors through time, as they mitigate the associated computational cost and stability/tuning issues [40]. This advantage is increased by their ideal parallelizability, and holds especially for long time-horizons [40].

Our novel cooperative coding scheme provides a potential explanation for the predominant excitation between highly similarly-tuned principal neurons observed in the cortex [43–47]: Its implementation minimizes (in balanced networks: lowers) the number of required synapses. This suggests that the number of synapses and thus space requirements are important constraints that shaped neural networks, giving a normative reason for the otherwise seemingly suboptimal like-to-like excitation.

Outlook We have analyzed cooperative coding in linear rate networks, where we quantified the trade-off between saving synapses and fast response times. As argued above, we generally expect our findings to extend to nonlinear and spiking neural networks. It will be important to study cooperative coding in these networks, especially in terms of their dynamics and connectivity. This will also allow to experimentally constrain neuronal and network parameters, gaining a more quantitative estimate of response time and synapse numbers.

For our linear rate networks, interpolating between the connectivities of the cooperatively coding and feedforward architectures yields hybrid models with the same stable steady state responses. In biological and spiking neural networks, additional, weak feedforward connections could be highly sparse (e.g., 10 % of the synapses of the feedforward network). Such sparse, 'anchoring' input would mean that recurrent amplification saturates at a subcritical value (e.g., 10-fold), implying also a convergence of response times for wide RFs. It would preserve the linear scaling of synaptic savings with RF size, and still agree with findings that the majority of input, both in terms of magnitude [81] and synapse number [138–140], is recurrent. Additionally, it would yield co-tuning of feedforward and recurrent inputs, as observed in experiments [77, 81, 141, 142]. Future research could verify these expectations.

Our analysis of WP and NP generates novel insights on their differential weight and learning dynamics, especially the accumulation of weight changes for synapses mediating weak input and the different dependence of convergence speed on batched trials. These may enable discovering and distinguishing experimental signatures of WP and NP. Also, it would be interesting to see if superstitious movements [143] can be interpreted and analyzed as 'irrelevant output components'.

We have shown that WP can be derived by applying the REINFORCE framework and decorrelating the resulting updates. This suggests the opportunity to modify the REINFORCE framework to yield unbiased updates also for correlated perturbations, allowing the discovery of an even greater family of learning rules.

Two attractive aspects of WP and NP are their biological plausibility and black-box character. It would

be most interesting to find ways to improve them using mechanisms available to biological neurons, thus trading the model-independence for better performance. For example, heterosynaptic plasticity might justify formulating dendrite-local instead of synapse-local learning rules. The co-tuning of close by synapses to large input components [144, 145] might be interpreted and used as a way to reduce irrelevant weights.

The credit assignment problem that WP and NP have to solve is complicated by the fact that they rely on a scalar reward signal, which provides very limited information. Promisingly, dopamine signals actually encode higher-dimensional feedback [146] and can be dispensed with a dynamic spatio-temporal profile [26, 147]. Delivering differential feedback to different parts of modularized networks that incorporate WP or NP may improve credit assignment if the feedback signals match the functionalities of the modules. Further, even random high-dimensional feedback may cause the targeted subpopulations to align with their rewarded function [148] in an effect similar to feedback alignment [34].

Finally, it would be important to study how WP and NP can interfere or synergize with activity-dependent plasticity, how they relate to reward-modulated Hebbian learning [117], and how WP and NP could be combined given that both neuronal activity and synaptic weights do fluctuate.

These are all exciting future research directions, and we hope our work cleared the path for their exploration.

APPENDIX A

Publication 1: Weight versus Node Perturbation Learning in Temporally Extended Tasks: Weight Perturbation Often Performs Similarly or Better

This appendix, 'publication 1', contains a full copy of the following article:

[1] **P. Züge**, C. Klos and R.-M. Memmesheimer
Weight versus Node Perturbation Learning in Temporally Extended Tasks: Weight Perturbation Often Performs Similarly or Better
Physical Review X **13** (2023) 021006

For the contribution statement, see Chapter 3.

Weight versus Node Perturbation Learning in Temporally Extended Tasks: Weight Perturbation Often Performs Similarly or Better

Paul Züge, Christian Klos, and Raoul-Martin Memmesheimer*

Neural Network Dynamics and Computation, Institute of Genetics, University of Bonn, Germany

(Received 9 December 2021; revised 9 September 2022; accepted 23 January 2023; published 11 April 2023)

Biological constraints often impose restrictions on plasticity rules such as locality and reward-based rather than supervised learning. Two learning rules that comply with these restrictions are weight (WP) and node (NP) perturbation. NP is often used in learning studies, in particular, as a benchmark; it is considered to be superior to WP and more likely neurobiologically realized, as the number of weights and, therefore, their perturbation dimension typically massively exceed the number of nodes. Here, we show that this conclusion no longer holds when we take two properties into account that are relevant for biological and artificial neural network learning: First, tasks extend in time and/or are trained in batches. This increases the perturbation dimension of NP but not WP. Second, tasks are (comparably) low dimensional, with many weight configurations providing solutions. We analytically delineate regimes where these properties let WP perform as well as or better than NP. Furthermore, we find that the changes in weight space directions that are irrelevant for the task differ qualitatively between WP and NP and that only in WP gathering batches of subtasks in a trial decreases the number of trials required. This may allow one to experimentally distinguish which of the two rules underlies a learning process. Our insights suggest new learning rules which combine for specific task types the advantages of WP and NP. If the inputs are similarly correlated, temporally correlated perturbations improve NP. Using numerical simulations, we generalize the results to networks with various architectures solving biologically relevant and standard network learning tasks. Our findings, together with WP's practicability, suggest WP as a useful benchmark and plausible model for learning in the brain.

DOI: 10.1103/PhysRevX.13.021006

Subject Areas: Biological Physics, Computational Physics, Statistical Physics

I. INTRODUCTION

Different, usually combined, strategies underlie the learning of tasks in humans and other animals [1,2]. Supervised learning allows large, rapid improvements. It is based on observing in which way an action is erroneous and on the ability of the nervous system to use this information for the improvement of neuronal dynamics in a directed manner. This may be implemented by translating an error vector into a vector of suitable synaptic weight updates [3]. Fast learning could be achieved by directly adapting the dynamics [4]. Reward-based learning (reinforcement learning), in contrast, uses only a scalar feedback signal. It is, thus, also applicable if errors are known with little specificity, for example, because there is sparse, delayed feedback about the cumulative effect of actions, which might tell only whether an action is erroneous but not how the generating neural activity can be improved.

A variety of models for reward-based learning have been developed in the context of theoretical neuroscience and machine learning [5,6]. Two conceptually straightforward implementations of such learning in neural networks are weight perturbation (WP) [7,8] and node perturbation (NP) [9,10]. Their underlying idea is to add perturbations to the weights or to the summed weighted inputs and to correlate them to the change of task performance. If the reward increases due to an attempted perturbation, the weights or the node dynamics are changed in its direction. If the reward decreases, the changes are chosen oppositely. NP and WP are used to model reward-based learning in biological neural networks, due to four properties [4,7,9–14]: (i) They are (with minor modifications) biologically plausible. (ii) They are applicable to a broad variety of networks and tasks. (iii) They are accessible to analytical exploration. (iv) They are optimal in the sense that the average of the generated weight change taken over all noise realizations is

*rm.memmesheimer@uni-bonn.de

along the reward gradient. The schemes' names were originally coined for approaches that directly estimate the individual components of the gradient using single perturbations to each weight or node [15].

WP explores with its random perturbations a space with dimensionality equal to the number of weights. For trials without temporal extent, NP needs only to explore a space with dimensionality equal to the number of nodes. The chain rule, amounting to a simple multiplication with the unweighted input strength, then allows one to translate a desirable change in the summed weighted inputs into a change in a particular weight strength. NP, thus, uses additional information on the structure of the network (namely, the linearity of input summation) to reduce the required exploration.

In linear approximation, the optimal direction of weight changes aligns with the direction of the gradient of the reward. WP and NP seemingly attempt to find this direction by trying out random perturbations. Since the dimension of the space of possible perturbation directions is large, the probability of finding the gradient is small and a lot of exploration is necessary. This impedes WP and NP. The number of weights and, thus, the dimensionality of the perturbation space searched by WP are much larger than the number of nodes. NP is, thus, considered more suitable for reward-based neural network learning [3,9,10,16] and its implementation in biological neural networks [17]. This is supported by quantitative analysis: Ref. [9] considers M linear perceptrons with N random inputs, using a student-teacher task. They find that for WP the optimal convergence rate of the student to the teacher weight matrix is by a factor NM worse than for exact gradient descent (GD). This is consistent with the argument that WP needs to search the NM-dimensional weight space to find the gradient, which is directly computed by GD. Accordingly, NP is worse than gradient descent by the dimensionality M of the node perturbation space.

The prerequisites of the arguments sketched above, however, do not hold in many biological situations. First, tasks in biology often extend in time and have a reward feedback that is temporally distant from the action [1,5,10,14,18]. Second, the effective dimension of neural trajectories and of learning tasks is often comparably low [19–21]. Our article analytically and numerically explores the perturbation-based learning of tasks with these features.

The article is structured as follows. First, we introduce the employed WP and NP learning models. We then derive analytic expressions for the evolution of expected error (negative reward) in linear networks solving temporally extended, low-dimensional reward-based learning tasks. This allows us to identify conditions for which WP outperforms NP as well as the underlying reasons. Furthermore, we delineate distinguishing qualitative characteristics of the weight and error dynamics. Finally, we numerically show that WP is comparably good or outperforms NP in different biologically relevant and standard network learning applications.

II. RESULTS

A. Learning models and task principles

Our study models the learning of tasks that are temporally extended. Time is split into discrete steps, indexed by $t = 1, \ldots, T$, where T is the duration of a trial. During this period, a neural network receives external input and generates output. At the end of a trial, it receives a scalar error feedback E about its performance [8,10,14,17,22]. To quantitatively introduce the learning rules, we consider a neuron i, which may be part of a larger network. It generates in the tth time bin an output firing rate z_{it}, in response to the firing rates r_{jt} of its N presynaptic neurons:

$$z_{it} = g\left(\sum_{j=1}^{N} w_{ij} r_{jt}\right). \tag{1}$$

Here, w_{ij} is the weight of the synapse (or the total weight of the synapses) from neuron j to neuron i. The generally nonlinear activation function g implements the relation between the total input current and the output firing rate of the neuron [5,23]. We note that the individual synaptic input currents $w_{ij} r_{jt}$ in the model sum up linearly. This is a standard assumption, and it is a requirement for the NP scheme [9,10,15,24]. In the presence of nonlinear dendritic compartments [25–27], each of these could be an independently perturbed node.

We model WP learning by adding in the beginning of a trial a temporally static weight change ξ_{ij}^{WP} to each of the weights w_{ij} [8,22]. The output of the neuron then reads

$$z_{it}^{\mathrm{pert,WP}} = g\left(\sum_{j=1}^{N} (w_{ij} + \xi_{ij}^{\mathrm{WP}}) r_{jt}^{\mathrm{pert,WP}}\right), \tag{2}$$

where $r_{jt}^{\mathrm{pert,WP}}$ are the input rates, which may have a perturbation due to upstream perturbed weights. ξ_{ij}^{WP} are independent and identically distributed (iid) zero-mean Gaussian white noise perturbations with standard deviation σ_{WP}, $\langle \xi_{ij}^{\mathrm{WP}} \xi_{kl}^{\mathrm{WP}} \rangle = \delta_{ik}\delta_{jl}\sigma_{\mathrm{WP}}^2$, where the angular brackets denote the average over perturbation realizations and δ is the Kronecker delta. The perturbations ξ_{ij}^{WP} change the output, which, in turn, influences the reward received at the end of the trial [Fig. 1(a), left-hand side]. We usually assume that the difference in reward between the perturbed and an unperturbed trial with $\xi_{ij}^{\mathrm{WP}} = 0$ for all i, j is used to estimate the gradient: When the reward increases, for small perturbations this means that the tried perturbation has a positive component along the reward gradient. Consequently, the update is chosen in the direction of that

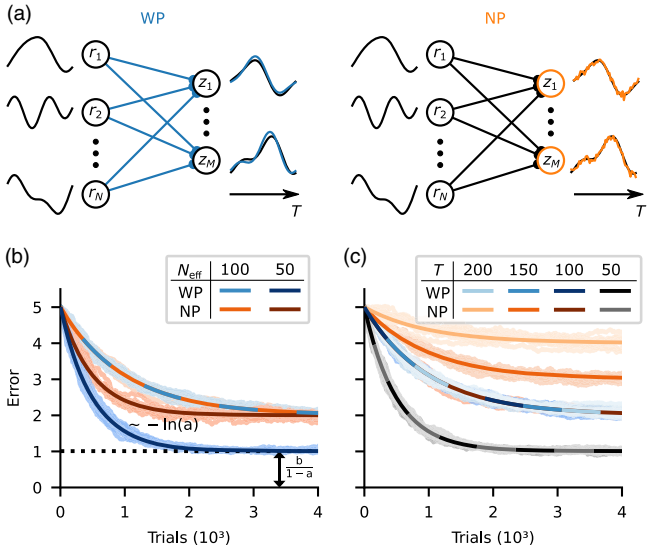

FIG. 1. Learning of temporally extended tasks in linear networks. (a) Schematic setup of WP and NP. The M outputs z_i are weighted sums of the N inputs r_j. Left: WP perturbs the weights at the beginning of a trial; the resulting perturbations of the weighted sums of the inputs and, thus, the outputs reflect the dimensionality and smoothness of the inputs (blue). Right: NP perturbs the weighted sums of the inputs with dynamical noise (orange). (b) WP (blue) works just as well as or better than NP (orange) when learning a single temporally extended input-output mapping. The error decay time decreases for WP and NP likewise with decreasing effective input dimension N_{eff} (light versus dark curves). In contrast, the residual error decreases only for WP. (c) Increased trial duration T does not change the progress of WP learning (blue curves lie on top of each other). In contrast, increasing T hinders NP learning by increasing the residual error (compare the increasingly lighter orange curves for larger T). If T decreases N_{eff} (gray curves), convergence is faster and to a lower residual error in both WP (because of the decrease in N_{eff}) and NP (because of the decrease in N_{eff} and T). (b) shows error curves from simulations (ten runs, shaded) together with analytical curves for the decay of the expected error (solid), for fixed $T, N = 100$, $M = 10$, and $N_{\mathrm{eff}} \in \{100, 50\}$; $\sigma_{\mathrm{eff}} = 0.04$. Theoretical curves and simulations agree well. For WP and $N_{\mathrm{eff}} = 50$, the decay rate $[-\ln(a)]$ and the residual error (dashed line) are highlighted. (c) shows error curves from simulations and theory similar to (b) for fixed $N = 100$ and $T \in \{200, 150, 100, 50\}$. N_{eff} is set to 100 but cannot be greater than T, such that $T = 50$ forces $N_{\mathrm{eff}} = 50$.

perturbation. When the reward decreases, the update is chosen in the opposite direction. We use the update rule

$$\Delta w_{ij}^{\mathrm{WP}} = -\frac{\eta}{\sigma_{\mathrm{WP}}^2}(E^{\mathrm{pert}} - E)\xi_{ij}^{\mathrm{WP}}, \tag{3}$$

where η is the learning rate, E^{pert} is the error of the perturbed trial, and E is the error of the unperturbed trial. For the delayed non-match-to-sample (DNMS) task, the error of the unperturbed trial is replaced by an average over the previous errors for biological plausibility. The proportionality of update size and obtained reward implies that, when averaging over the distribution of the perturbations, the weight change

$$\langle \Delta w_{ij}^{\mathrm{WP}} \rangle \approx -\eta \frac{\partial E}{\partial w_{ij}} \tag{4}$$

is parallel to the reward gradient [Eq. (A2)]. Since $\langle \xi_{ij}^{\mathrm{WP}} \rangle = 0$, this holds for any baseline in Eq. (3). The employed choice of baseline E guarantees that for small perturbations the weight change has a positive component in the direction of the reward gradient and, thus, always reduces the error for sufficiently small learning rate η [8]. In fact, it minimizes the update noise, i.e., the fluctuations of updates around the gradient [Eq. (A5)].

WP treats the system as a black box, mapping parameters w onto a scalar error function E. In other words, it uses the information that the weights are fixed parameters during a trial but does not take advantage of specifics of the network structure. This is in contrast to NP, which takes into account some minimal structural knowledge, namely, the linear summation of input currents, but not the constancy of the weights.

Instead of perturbing the weights directly, NP adds noise to the sum of the inputs:

$$z_{it}^{\mathrm{pert,NP}} = g\left(\sum_{j=1}^{N} w_{ij} r_{jt}^{\mathrm{pert,NP}} + \xi_{it}^{\mathrm{NP}}\right) \tag{5}$$

[9,10] [see Fig. 1(a), right-hand side]. ξ_{it}^{NP} are iid zero-mean Gaussian white noise perturbations with standard deviation σ_{NP}, $\langle \xi_{it}^{\mathrm{NP}} \xi_{ms}^{\mathrm{NP}} \rangle = \delta_{im}\delta_{ts}\sigma_{\mathrm{NP}}^2$. We note that for temporally extended tasks, in contrast to WP, the noise must be time dependent to explore the space of time-dependent sums of inputs [10]. For temporally constant noise, only the temporal mean of the total input would be varied and, thus, improved.

The NP update rule can be defined as

$$\Delta w_{ij}^{\mathrm{NP}} = -\frac{\eta}{\sigma_{\mathrm{NP}}^2}(E^{\mathrm{pert}} - E)\sum_{t=1}^{T} \xi_{it}^{\mathrm{NP}} r_{jt} \tag{6}$$

[10], with the eligibility trace $\sum_{t=1}^{T} \xi_{it}^{\mathrm{NP}} r_{jt}$. As for WP, this yields an average weight update parallel to the reward gradient, which again holds for any baseline of the weight update. The choice of baseline E again minimizes the update noise [Eq. (A6)].

The NP update rule effectively incorporates an error backpropagation step, which reduces to a simple multiplication with r_{jt} due to the linearity of the spatial synaptic input summation. This allows one to perturb only summed inputs instead of individual weights and may be

expected to increase the performance of NP compared to WP [3,10,14,17].

B. Theoretical analysis

We analytically compare WP and NP for temporally extended tasks by training a set of M linear perceptrons with N inputs. The task is to learn the mapping of a single fixed input sequence of duration T to a target output sequence in a reward-based manner. The task choice is motivated first by biological motor tasks that require such a mapping, like the learning of their single song in certain songbirds (see Sec. III). Second, it yields novel insights, as it is the opposite extreme case to having no time dimension and different, random inputs in each trial; this case is treated analytically by Ref. [9] (see the introduction). Third, our findings yield an understanding of the learning performance for more general temporally extended tasks and networks studied later in this article. The analysis shows how learning depends on task dimensions and the structure of the input. Furthermore, it reveals specific disadvantages of WP and NP. Importantly, our theoretical considerations cover very general sequences. In particular, they hold for sequences with and without correlations between subsequent inputs. Furthermore, the sequences can be arbitrarily reordered, also differently in each trial. They may, therefore, also be interpreted as sets or batches of inputs. In Sec. II F, we consider temporally correlated sequences, for which such an interpretation is not useful anymore. In Secs. II G and II I, we relax the assumption of exactly repeated input sets.

The perceptrons generate as outputs the product of their $M \times N$ weight matrix w with the inputs

$$z_{it} = \sum_{j=1}^{N} w_{ij} r_{jt}, \tag{7}$$

where $i = 1, \ldots, M$ [Eq. (1) and Fig. 1(a)]. For now, we assume that the target output can be produced with target weights w^*, that is, $z^*_{it} = \sum_{j=1}^{N} w^*_{ij} r_{jt}$. This condition is alleviated in Sec. II E. The learning goal is to reduce the quadratic deviation of each output from its target, which can be expressed through the weight mismatch $W = w - w^*$ and the input correlation matrix $S_{jk} = (1/T) \sum_{t=1}^{T} r_{jt} r_{kt}$ [5]:

$$E = \frac{1}{2T} \sum_{i=1}^{M} \sum_{t=1}^{T} (z_{it} - z^*_{it})^2 = \frac{1}{2} \mathrm{tr}[WSW^T]. \tag{8}$$

We note that with this quadratic error function the average weight update [cf. Eq. (4)] follows the gradient exactly, for both WP and NP [Eqs. (A16) and (A17)].

We assume that the inputs are composed of N_{eff} orthogonal latent inputs; all other input components are zero (this is relaxed in Sec. II I). Since there are at most T linearly independent vectors of length T, the effective input dimension N_{eff} is bounded by $N_{\mathrm{eff}} \leq T$. $T > 1$ thus renders our learning problem nontrivial, by allowing for inputs that are higher dimensional when considering the input-output relation of a single sequence. In biological systems, inputs are low dimensional; N_{eff} is often of the order of 10 (Sec. III), in particular, $N_{\mathrm{eff}} \ll N$. As long as inputs are summed linearly, for clarity we then hypothetically "rotate" the inputs such that only the first N_{eff} inputs are nonzero and equal to the latent ones [Fig. 2(a)]. This allows us to speak about relevant and irrelevant inputs instead of relevant (nonzero) and irrelevant (zero) input space directions. It does not affect the WP or NP learning process, because all perturbations are isotropic and the error function is rotationally invariant [Supplemental Material, Eq. (S1) [28]]. In other words, all results, in particular, the dynamics of the error decay, hold identically for the original networks with nonrotated inputs where all actual inputs may be nonzero. For simplicity in our mathematical analysis, we assume that all latent inputs have the same strength α^2, i.e., $(1/T) \sum_{t=1}^{T} r_{it}^2 = \alpha^2$ for the nonzero inputs $i = 1, \ldots, N_{\mathrm{eff}}$. A partial treatment of networks with inhomogeneous latent input strength is given in Supplemental Material Sec. IV [28].

C. Error dynamics

To elucidate the learning process and its dependence on the network and task parameters, we analytically derive the evolution of the expected error. This requires the computation of the error signal $E^{\mathrm{pert}} - E$ and weight update after a given perturbation to determine the new error. Subsequent averaging over all perturbations yields the expected error at trial n, $\langle E(n) \rangle$, as a function of $\langle E(n-1) \rangle$, specifically, a linear recurrence relation

$$\langle E(n) \rangle = a \langle E(n-1) \rangle + b \tag{9}$$

(see Appendix B for the detailed derivation). The speed of learning is determined by the convergence factor a, while the per-update error increase b limits the final performance. Learning stops at a finite error when an equilibrium between gradient-related improvement and reward noise-induced deterioration is reached. The recurrence relation is solved by

$$\langle E(n) \rangle = [\langle E(0) \rangle - E_f] a^n + E_f. \tag{10}$$

For $a < 1$, the average error $\langle E(n) \rangle$ converges exponentially at a rate $-\ln(a)$ toward a finite final (residual) error of $E_f = b/(1-a)$, as shown in Fig. 1(b). Usually, in our settings a is sufficiently close to 1 to well approximate the convergence rate by $-\ln(a) \approx 1 - a$.

To understand how learning depends on the task parameters, we first consider the speed of learning. The determining convergence factor

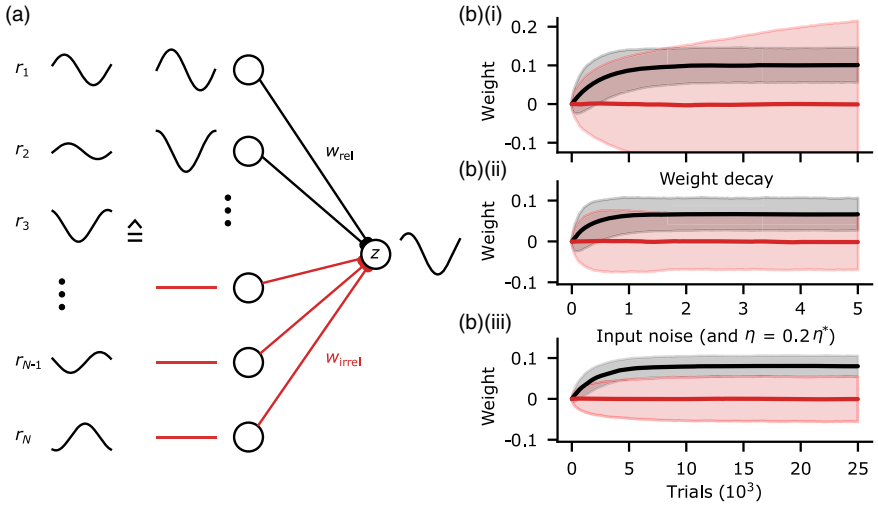

FIG. 2. Hypothetical rotation of inputs and weight diffusion. (a) Because the inputs (left, black) are summed linearly, they can be "rotated" so that for our tasks the first N_{eff} inputs are nonzero and agree with the latent inputs (middle, black). The remaining inputs are then zero (middle, red), and their weights are irrelevant for the output (right, red). (b)(i) In WP with finite perturbation size σ_{WP}, the irrelevant weights diffuse without bounds (red), while the relevant weights converge and fluctuate (black) around the teacher weights. Displayed are the mean (solid) and standard deviation (shaded area) of the weight ensembles. (b)(ii) Weight decay or (b)(iii) input noise confines the diffusion. In (a), $N_{\mathrm{eff}} = 2$, the latent inputs are a sine and a cosine. Parameters in (b)(i) and (b)(ii): $M = 10, N, T = 100$, $N_{\mathrm{eff}} = 50$, $\sigma_{\mathrm{eff}} = 0.04$, teacher weights $w_{\mathrm{rel},i} = 0.1$, and weight decay $\gamma_{\mathrm{WD}} = 0.999$; results are averaged over ten runs. (b)(iii) The same parameters, except $\eta = 0.2\eta^*$ and added iid white input noise with strength $\sigma^2_{\mathrm{noise}} = 0.5$ (SNR $= 2$).

$$a = 1 - 2\eta\alpha^2 + \eta^2\alpha^4(MN_{\mathrm{eff}} + 2) \tag{11}$$

[Eq. (B39)] is affected by two opposing effects: On average, updates follow the gradient, thus reducing the error. This is reflected by a reduction of a by $-2\eta\alpha^2(+\eta^2\alpha^4)$, responsible for convergence. However, updates fluctuate, adding a diffusive part to the weight evolution which slows convergence down. Although these fluctuations, having zero mean, do not influence the expected error to linear order, they do so quadratically. Thus, their contribution to a, $\eta^2\alpha^4(MN_{\mathrm{eff}} + 1)$, is quadratic in the learning rate η. It is approximately proportional to MN_{eff}, the number of relevant weights that read out from nonzero inputs: Fluctuations in each of these weights yield the same contribution—the exception being the twice as strong fluctuations along the single gradient-parallel direction, which together with the quadratic effect of the mean update cause the $+2$ in Eq. (11).

The fluctuations originate from a credit assignment problem: Only the perturbation parallel to the error gradient can be credited for causing the linear part of the error signal $E^{\mathrm{pert}} - E$. WP has no way of directly solving the credit assignment problem of identifying this direction. Thus, the perturbations of all MN weights are equally amplified in the constructions of their updates [Eq. (3)] such that all weights fluctuate. This entails fluctuations in the MN_{eff} relevant weights, which influence output and error. NP can at least partially solve the credit assignment problem by using eligibility traces, which are zero for weights that read out from zero inputs. By projecting each of its (M) T-dimensional output perturbations onto the effectively N_{eff}-dimensional inputs, NP restricts its updates to the MN_{eff}-dimensional subspace of relevant weights. The convergence speed, thus, becomes independent of T as for WP. Interestingly, WP and NP therefore converge at the same speed despite their different numbers of fluctuating weights. The reason is that the fluctuations of the relevant weights are the same for both algorithms.

The balance between the improvement resulting from following the gradient $(\sim\eta)$ and the deterioration due to the fluctuations of relevant weights $(\sim\eta^2)$ in Eq. (11) is controlled by the learning rate: Small learning rates imply averaging out fluctuations over many updates and, therefore, dominance of gradient following, leading to convergence. For the remainder of the analysis of this setting, both algorithms are compared at their optimal learning rate η^*, which is defined to yield fastest convergence, in other words, to minimize a. This definition is chosen because it is conceptually straightforward, and Eq. (11) directly leads to the simple expressions

$$\eta^* = \frac{1}{(MN_{\mathrm{eff}} + 2)\alpha^2}, \qquad a^* = 1 - \frac{1}{MN_{\mathrm{eff}} + 2}. \tag{12}$$

Here, the factor $1/\alpha^2$ in η^* cancels the scaling of the gradient with the input strength and equals the optimal learning rate for GD [Eq. (B5)]. In order to allow for averaging out the update fluctuations, WP and NP learning additionally have to slow down by a factor of

approximately MN_{eff}. Learning diverges for $\eta \to 2\eta^*$ where $a \to 1$. Equation (12) shows that WP's convergence rate is worse than GD's by a factor generally smaller than the number of weights. Further, NP's convergence rate is worse by a factor generally larger than the number of nodes. Thus, the number of weights or nodes is insufficient to predict the performance of WP or NP, respectively.

The per-update error increase and the final error, b and E_f, result from finite perturbation sizes. Finite perturbation sizes lead, due to the curved, quadratic error function, to an estimate that is at least slightly incompatible with the linear approximation assumed by the update rules [cf. Eq. (4)]. This is particularly apparent when the output error and, thus, the gradient are (practically) zero: Any finite weight or node perturbation then leads to an increase of the error and, thus, to an opposing weight update instead of no weight modification. This prevents the weights from reaching optimal values and results in a finite final error E_f. The described difference between perturbation-based error estimate and linear approximation is a form of "reward noise." It is nonzero only for finite perturbation size, as reflected by the dependence of b and E_f on σ (which is quadratic due to the quadratic error nonlinearity).

For a fair comparison of WP and NP, we choose σ_{WP} and σ_{NP} such that they lead to the same effective perturbation strength σ^2_{eff}, as measured by the total induced output variance. This leads to $\sigma^2_{\text{NP}} = \sigma^2_{\text{eff}}$ and $\sigma^2_{\text{WP}} = 1/(\alpha^2 N_{\text{eff}}) \cdot \sigma^2_{\text{eff}}$ [Eq. (A22)]. Evaluated at the optimal learning rate η^*, the leading-order term of the final error is

$$E_f^{\text{WP}} = \frac{b^*_{\text{WP}}(\eta^*)}{1-a^*} \approx \frac{1}{8}\sigma^2_{\text{eff}} \cdot M^2 N_{\text{eff}}, \tag{13}$$

$$E_f^{\text{NP}} = \frac{b^*_{\text{NP}}(\eta^*)}{1-a^*} \approx \frac{1}{8}\sigma^2_{\text{eff}} \cdot M^2 T. \tag{14}$$

Importantly, the final error of WP is here generally smaller, by a factor $N_{\text{eff}}/T \leq 1$. To understand this, we focus for both WP and NP on the output perturbations that they generate. By perturbing the weights, WP induces output perturbations that are linear combinations of the inputs. These are confined to the effectively MN_{eff}-dimensional subspace in which also the (realizable part of the) output error gradient $(z - z^*)/T$ lies. NP, on the other hand, creates an entirely random MT-dimensional perturbation vector [Fig. 1(a)]. Only the projection of this vector onto the output gradient is useful for learning. This projection is smaller for NP's random vector, since the vector has effectively a larger dimensionality than the output perturbation vector of WP, at the same length. NP compensates this deficit by amplifying the smaller gradient projection more strongly. It, thus, achieves the same mean update and convergence speed as WP. However, it also more strongly amplifies the reward noise that comes with larger perturbation sizes, which results in a larger final error. The scaling of E_f with $M^2 N_{\text{eff}}$ or $M^2 T$ reflects the effective output perturbation dimensions, MN_{eff} or MT of WP or NP, and additionally the general scaling of errors with M [Eq. (8), Supplemental Material Sec. I, and Eqs. (S55)–(S57) [28]].

Taken together, we observe that here WP learning works just as well as or better than NP. Both algorithms have the same speed of convergence, but the final error E_f of WP is smaller than or equal compared to NP. The rate of convergence decreases with increasing M and N_{eff}. Longer trial durations T harm NP by linearly increasing E_f. Larger effective input dimensionality N_{eff} similarly harms WP. This result differs from the observation in Ref. [9] that WP converges much (N-times) slower than NP. The reason is that our networks learn a single temporally extended input-output relation, while those in Ref. [9] learn the weights of a teacher network, by trials with random input of duration $T = 1$. We explore the relation between the results in detail in Sec. II H.

D. Weight diffusion

When the input has less than maximal dimensionality, $N_{\text{eff}} < N$, only certain combinations of weights read out nonzero components of the input. This becomes particularly clear for the considered rotated inputs: If WP adds a perturbation to a weight mediating zero input, to an irrelevant weight, the output and the error remain unchanged. This missing feedback leads to an unbounded diffusionlike random walk of irrelevant weights. For unrotated inputs, the weight strength diffuses in irrelevant weight space directions. We see below (Sec. II G) that the weight diffusion harms performance when learning multiple input-output patterns.

We find that, for WP in the limit of infinitesimally small perturbations $\sigma_{\text{WP}} \to 0$, all weights initially change and then converge (Supplemental Material, Fig. S1 [28]). This is because the learning-induced drift of relevant and the diffusion of irrelevant weights both stop when the error converges to zero: The error E is quadratic, such that for infinitesimally small perturbations $E^{\text{pert}} = E$ at its minimum. In contrast, for finite perturbations a residual error remains and weights continue to change. In particular, irrelevant weights continue to diffuse [Fig. 2(b)(i)]. The quantitative details of the weight diffusion process can be analytically understood [Supplemental Material Sec. II, Eqs. (S61) and (S64) [28]]. Standard mechanisms such as an exponential weight decay [29,30] confine their growth. Simultaneously, they bias the relevant weights toward zero and therewith increase the residual error [Fig. 2(b)(ii)]. Also, input noise confines the weight spread, by adding error feedback to irrelevant weights. Simultaneously, the noise increases the final task performance error and enforces a lower learning rate; see Fig. 2(b)(iii), Sec. II I, and Figs. 5(a) and 5(b).

NP does not generate weight diffusion in noise-free networks: The rotated inputs make it obvious that in NP the eligibility trace [Eq. (6)] selects only the weights from

relevant inputs to be updated, since for irrelevant inputs we have $r_{jt} = 0$ for all t such that $\Delta w_{ij}^{\mathrm{NP}} = 0$. Input noise renders irrelevant inputs and their weight updates nonzero, such that irrelevant weights diffuse also for NP (Sec. II I).

Differences in weight spread and updates between WP and NP suggest experimental measurements to distinguish which one of them underlies learning of a certain task: As long as the noise is weak compared to the signal such that the task can be satisfied with high precision, the spread of irrelevant weights is with NP much smaller than with WP [Fig. 5(b)]. Furthermore, a large variance in the weight updates that is independent of presynaptic activity together with a resulting weight spread that is largest for weight directions that read out weak latent inputs point to WP [Fig. 5(b) and Supplemental Material Sec. IV, Eq. (S126) [28]]. Consistent with this, prominent random walklike weight changes that are unrelated to neuronal activity and task learning are common in biological neural networks [31]. Weight updates whose variance scales with input strength but whose final spread is independent of it [Fig. 5(b) and Supplemental Material Sec. IV, Eq. (S128) [28]] point to NP.

E. Unrealizable targets

In the previous sections, we assume that the target outputs z_{it}^* could be exactly realized by setting the perceptron weights w equal to some target weights w^*, $z_{it}^* = \sum_{j=1}^{N} w_{ij}^* r_{jt}$. In general, however, the target outputs may contain components d that cannot be generated by the network, which is limited to producing linear combinations of the inputs. Unrealizable components are orthogonal to all inputs when interpreted as T-dimensional vectors, $\sum_{t=1}^{T} d_{it} r_{jt} = 0 \;\forall\, i, j$. The target may be written as a sum of realizable and orthogonal unrealizable parts, $z_{it}^* = \sum_{j=1}^{N} w_{ij}^* r_{jt} + d_{it}$. An illustration of such a target is given in Fig. 3(a). In practice, unrealizable targets occur, for example, in machine learning classification tasks, see Sec. II N.

WP induces output perturbations $\delta z_{it} = \sum_{j=1}^{N} \xi_{ij}^{\mathrm{WP}} r_{jt}$, which are linear combinations of the inputs. The components of $z^{\mathrm{pert}} - z^*$ that are orthogonal to all inputs, d, thus always remain unchanged, irrespective of the current student weights and applied perturbations. This leads to the same constant additive contribution $E_{\mathrm{opt}} = 1/(2T)\mathrm{tr}[dd^T]$ to the perturbed and unperturbed errors E^{pert} and E [Eq. (8)]. It cancels in the weight update rule [Eq. (3)] such that WP learning is unchanged and Eq. (10) still holds when shifting its final error to $E_{f,\mathrm{unr}}^{\mathrm{WP}} = E_f^{\mathrm{WP}} + E_{\mathrm{opt}}$ [Eq. (S119) [28]]. E_{opt} marks the minimum error that necessarily remains even with $w = w^*$, due to the unrealizable components.

In contrast, NP perturbs the outputs with white noise. This noise generally has a nonzero component along d, which affects E_{pert}. Since such a component cannot be realized through an update of the weights, the resulting change of the error is noninstructive and represents reward noise that adds noise to the updates. Consequently, while the convergence factor a remains unchanged, the final error of NP increases more strongly than for WP [Fig. 3(b)]. At the optimal learning rate, the increase in final error due to unrealizable target components is twice that of WP: $E_{f,\mathrm{unr}}^{\mathrm{NP}} \approx E_f^{\mathrm{NP}} + 2E_{\mathrm{opt}}$ [Supplemental Material Sec. I, Eq. (S58) [28]]. For $E_{\mathrm{opt}} > E_f^{\mathrm{NP}}$, the coupling of node perturbations to unrealizable target components becomes NP's main contribution to the part of the final error that exceeds the unavoidable E_{opt}.

F. Input and perturbation correlations

Our results hold for very general sequences. In particular, correlations in the input may be present or absent without affecting learning. Furthermore, the T input-output relations can be temporally permuted. These invariances follow

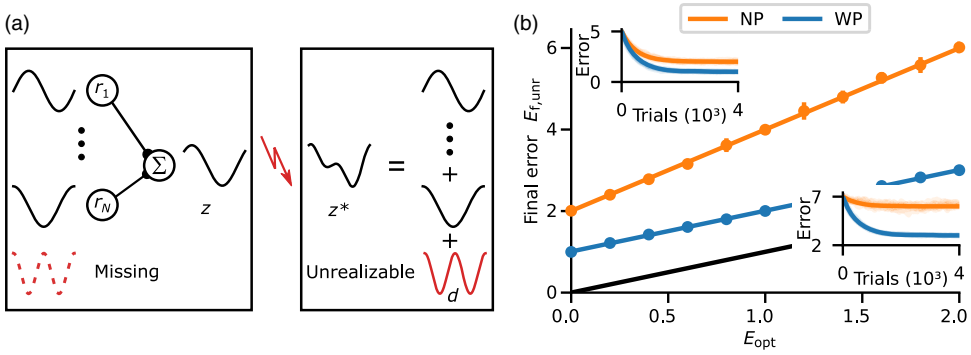

FIG. 3. Unrealizable target components harm NP learning. (a) General targets may contain a component that is perpendicular to any input and, thus, unrealizable (red). (b) Final error after convergence as a function of the error E_{opt} that necessarily remains, since the target is unrealizable. The final error of WP (blue) is shifted only by E_{opt}; that of NP (orange) increases twice as fast, by approximately $2E_{\mathrm{opt}}$. Data points: mean and standard deviation (averaged over ten simulated runs) of the final error. Curves: theoretical predictions. Black: E_{opt}. Insets: error dynamics for $E_{\mathrm{opt}} = 0$ (left) and $E_{\mathrm{opt}} = 2$ (right). Parameters: $M = 10$, $N, T = 100$, $N_{\mathrm{eff}} = 50$, and $\sigma_{\mathrm{eff}} = 0.04$.

straightforwardly from the weight update equations (Supplemental Material Sec. V [28]). We note that overly strong input correlations reduce the effective temporal dimension of the input such that its effective dimension cannot be kept up [cf. Fig. 1(c)], which affects learning.

WP generates output perturbations that are automatically adapted to the inputs (Sec. II E). To adapt NP to tasks with temporal input correlations, we modify it to time-correlated NP, NPc. We generate the correlated perturbations by temporal low-pass filtering of white noise, with filtering time constant $\tau_{\mathrm{corr}}^{\mathrm{pert}}$. Other possibilities are to compose them from low-frequency Fourier modes or to use perturbations that are piecewise constant. As the correlation of the perturbation decays during $\tau_{\mathrm{corr}}^{\mathrm{pert}}$, for a reasonable representation of the perturbation we need to specify it at $T_{\mathrm{eff}}^{\mathrm{pert}} = T/(\tau_{\mathrm{corr}}^{\mathrm{pert}}+1)$ time points with temporal distance $\tau_{\mathrm{corr}}^{\mathrm{pert}}+1$. For vanishingly short correlations, $\tau_{\mathrm{corr}}^{\mathrm{pert}}=0$, $T_{\mathrm{eff}}^{\mathrm{pert}}=T$ and NPc equals NP; a large filtering time constant generates perturbations that vary slowly and have small effective temporal dimension $T_{\mathrm{eff}}^{\mathrm{pert}}$.

If the inputs have similar temporal correlations, the filtering of the perturbation concentrates its perturbative power on the realizable output subspace, since this is spanned by the inputs. This reduces the update noise due to the quadratic reward noise from finite perturbation sizes [cf. Eq. (S112) [28]], because perturbations that are more aligned with the output gradient require less amplification to yield a sizable expected update along the weight gradient. Furthermore, it reduces the linear reward noise resulting from coupling to unrealizable target components [cf. Eqs. (S108) and (S80) [28]]. Both noise reductions lower the final error. We find that NPc robustly improves upon NP over a range of filtering time constants similar to that of the inputs (Fig. 4). If the perturbations become too smooth, however, they also lose power in the realizable output subspace, which slows the learning of realizable target components with higher frequencies down. This is because the suppressed modes with their comparably small amplitude contribute little to the projection of the perturbations onto the T-dimensional output gradients (despite being prominent in the latter), which results in a small contribution to the error signal [Eq. (S3) [28]]. This contribution, in turn, determines the magnitude of the mean weight update used to match an output mode to the target. Consequently, the update part used to match the high-frequency modes is small, and the matching takes a

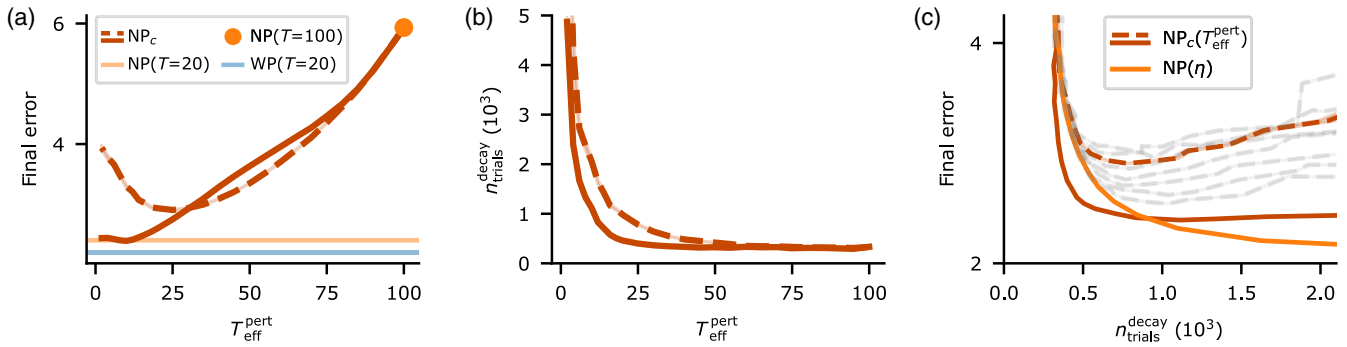

FIG. 4. Temporally correlated perturbations improve NP if the input has similar correlations. (a) Final error of NPc versus the effective time dimension $T_{\mathrm{eff}}^{\mathrm{pert}}$ of its perturbations; smaller $T_{\mathrm{eff}}^{\mathrm{pert}}$ means smoother perturbations. The inputs are constructed with $T_{\mathrm{eff}}^{\mathrm{input}}=20$ (red continuous line) or $T_{\mathrm{eff}}^{\mathrm{input}}=100$ (uncorrelated, red dashed line). The final error of NPc decreases compared to that of NP (orange dot) for $T_{\mathrm{eff}}^{\mathrm{pert}}<T=100$. For correlated inputs and $T_{\mathrm{eff}}^{\mathrm{pert}}\leq T_{\mathrm{eff}}^{\mathrm{input}}=20$, NPc's final error reaches that of NP in a reduced task with $T=20$ (orange line; WP, blue line); for uncorrelated inputs and small $T_{\mathrm{eff}}^{\mathrm{pert}}$, it increases again. This increase is not caused by lack of convergence. (b) Convergence time, measured as the number of trials until 95% of the final error correction is reached, versus $T_{\mathrm{eff}}^{\mathrm{pert}}$. Learning gets considerably slower if the correlation times of the perturbation are longer than those of the input, i.e., $T_{\mathrm{eff}}^{\mathrm{pert}}<T_{\mathrm{eff}}^{\mathrm{input}}$. The same curve markings as in (a). (c) Simultaneous plot of error and convergence time of NPc [red line, data as in (a) and (b) for correlated (solid line) or uncorrelated input (dashed line); thus, curves are obtained by varying $T_{\mathrm{eff}}^{\mathrm{pert}}$; faint gray line, curves for uncorrelated input with different learning rates η] and NP (orange, curves obtained by varying the learning rate η). For correlated inputs and similarly correlated perturbations, NPc yields a true improvement over NP: It has simultaneously a smaller error and smaller convergence time. For considerably longer correlation times, the final error saturates, but the convergence time increases (region with $n_{\mathrm{trials}}^{\mathrm{decay}}>1000$). NP with reduced learning rate η there yields a smaller final error at the same convergence time. For uncorrelated input, NPc does not yield a true improvement. In particular, the smaller final error (a) can be achieved at smaller convergence times by NP with reduced η, even if NPc is allowed to additionally adapt its learning rate (gray curves). Parameters: $N_{\mathrm{eff}}=M=10$, $N=T=100$, $\sigma_{\mathrm{eff}}=0.04$, $E_{\mathrm{opt}}=2$, and 20000 trials. (a),(b) Mean and standard error of the mean (SEM, partly occluded). (c) Mean values and SEM of final error (red and orange lines). In gray curves, η is modified relative to the red curve by a factor ranging from 0.5 (lowest curve) to 1.2 (highest curve) in steps of 0.1.

long time. With optimal perturbation correlation time $\tau_{\mathrm{corr}}^{\mathrm{pert}} \approx \tau_{\mathrm{corr}}^{\mathrm{input}}$ (Sec. IV), NPc approaches the performance (as measured by final error and convergence speed) of NP on a substitute task with $T_{\mathrm{eff}}^{\mathrm{input}}$ bins; see Figs. 4(a) and 4(b). This indicates that our analytical considerations for NP transfer to those of NPc with optimally correlated noise if we take into account that the correlations effectively reduce the trial duration to $T_{\mathrm{eff}}^{\mathrm{input}}$. In particular, in Fig. 4, optimal NPc performs worse than WP due to the low dimensionality and the remaining temporal extension of the task as well as the unrealizable target components.

G. Multiple subtasks

In general learning tasks, inputs and targets may vary from trial to trial. To obtain an intuition for how this affects the speed of WP and NP learning, we here consider a simplified case: The goal is to solve a task with an overall effective input dimension of $N_{\mathrm{eff}}^{\mathrm{task}}$. The task has the same properties as the tasks before where each trial was identical. In particular, it has $N_{\mathrm{eff}}^{\mathrm{task}}$ orthogonal latent inputs of strength α^2, and the inputs are rotated such that only the first $N_{\mathrm{eff}}^{\mathrm{task}}$ inputs are nonzero. The task is, however, not presented as a whole, but in pieces: In each trial, a random subset of $N_{\mathrm{eff}}^{\mathrm{trial}}$ out of the first $N_{\mathrm{eff}}^{\mathrm{task}}$ inputs are active to train the network. The error in an individual trial then depends only on its $MN_{\mathrm{eff}}^{\mathrm{trial}}$ trial-relevant weights, while the performance on the full task depends on the $MN_{\mathrm{eff}}^{\mathrm{task}}$ task-relevant weights. The ratio $N_{\mathrm{eff}}^{\mathrm{task}}/N_{\mathrm{eff}}^{\mathrm{trial}} = P$ marks the number of trials needed to gather information on all task-relevant weights.

NP updates only the weights relevant in a trial (Sec. II D). Also, for tasks consisting of multiple subtasks, it can thus operate at the learning rate that is optimal for a trial, $\eta_{\mathrm{NP}}^{*} = 1/[(MN_{\mathrm{eff}}^{\mathrm{trial}} + 2)\alpha^2]$ [cf. Eq. (12)]. Because an update improves only $MN_{\mathrm{eff}}^{\mathrm{trial}}$ of the $MN_{\mathrm{eff}}^{\mathrm{task}}$ task-relevant weights, the convergence rate $-\ln a \approx 1 - a$ of the expected error, averaged over the input distribution, is smaller by a factor of $1/P$ than for a single input pattern [Supplemental Material Sec. III, Eq. (S79) [28]]:

$$a_{\mathrm{NP}}^{*} = 1 - \frac{1}{P}\frac{1}{MN_{\mathrm{eff}}^{\mathrm{trial}} + 2}. \tag{15}$$

WP, on the other hand, updates all weights such that the weights that are irrelevant for the trial are changed randomly (Sec. II D). This worsens the performance for the inputs of other trials. Because there are now $MN_{\mathrm{eff}}^{\mathrm{task}}$ task-relevant weights whose fluctuations hinder learning, WP has an optimal learning rate of only $\eta_{\mathrm{WP}}^{*} = 1/[(MN_{\mathrm{eff}}^{\mathrm{task}} + 2)\alpha^2]$. As for NP, each trial's progress is only on $1/P$ of the task-relevant weights, such that the optimal convergence factor for WP on the full task is [Supplemental Material Sec. III, Eq. (S78) [28]]

$$a_{\mathrm{WP}}^{*} = 1 - \frac{1}{P}\frac{1}{MN_{\mathrm{eff}}^{\mathrm{task}} + 2}. \tag{16}$$

The convergence of WP is, thus, slower than that of NP by roughly $1/P$, the ratio of $N_{\mathrm{eff}}^{\mathrm{task}}$ and $N_{\mathrm{eff}}^{\mathrm{trial}}$ [Fig. 6(c)].

Our results have concrete implications for learning of multiple actions such as sequences of movements [32]. They can be learned by splitting them into subsets, which are called (mini)batches in machine learning. If we assume for simplicity that individual data points are pairwise orthogonal and have no time dimension, each batch corresponds in our terminology to a subtask, the number of batches to P, the dimensionality of the input data to $N_{\mathrm{eff}}^{\mathrm{task}}$, and the batch size N_{batch} to $N_{\mathrm{eff}}^{\mathrm{trial}}$. For $MN_{\mathrm{eff}}^{\mathrm{trial}} \gg 2$, Eqs. (15) and (16) thus imply that the convergence rate of NP is independent of the batch size while that of WP is proportional to the batch size and reaches NP's convergence rate for full batch learning (Supplemental Material, Fig. S5 [28]). The same holds for the optimal learning rates as α^2 scales inversely with the batch size [Supplemental Material Sec. III, Eqs. (S90), (S92), and (S93) [28]].

H. Comparison with Ref. [9]

Reference [9] investigates how a student network learns the responses of a teacher network to arbitrary input with GD, NP, and WP, using patterns without temporal extent. In contrast to our tasks with typically $N_{\mathrm{eff}} < N$ or $N_{\mathrm{eff}}^{\mathrm{task}} < N$, successful learning in Ref. [9] requires one to match all weights of the teacher network. In other words, the student network is trained at its capacity limit, where (only) one weight configuration fulfills the task. It learns from random input patterns and the teacher's responses to them. This is a special case of the setup introduced in Sec. II G, where (i) the task dimension equals the input dimension, $N_{\mathrm{eff}}^{\mathrm{task}} = N$ (since the employed random input patterns lie in arbitrary directions of input space), and (ii) there is no temporal extent of the tasks, $T = 1$. The latter implies that a single input pattern has effective dimension $N_{\mathrm{eff}}^{\mathrm{trial}} = 1$: All N inputs in a single pattern are linearly dependent, since they are scalar, temporal vectors of length 1 (we could rotate the input pattern such that it has only one nonzero entry).

A comparison of our results for the convergence speed in the described special case with those of Ref. [9] reveals that they agree approximately when straightforwardly setting $N_{\mathrm{eff}}^{\mathrm{trial}} = 1$, $N_{\mathrm{eff}}^{\mathrm{task}} = N$, and $P = N_{\mathrm{eff}}^{\mathrm{task}}/N_{\mathrm{eff}}^{\mathrm{trial}} = N$ in Eqs. (15) and (16):

Algorithm	Our results	Results from Ref. [9]
WP	$a_{\mathrm{WP}}^{*} = 1 - \frac{1}{N}\frac{1}{MN+2}$	$a_{\mathrm{WP}}^{*} = 1 - \frac{1}{N+2}\frac{1}{MN+2}$
NP	$a_{\mathrm{NP}}^{*} = 1 - \frac{1}{N}\frac{1}{M\cdot 1+2}$	$a_{\mathrm{NP}}^{*} = 1 - \frac{1}{N+2}\frac{1}{M+2}$.

The reason for the remaining difference is that the individual inputs in Ref. [9] are drawn from a Gaussian distribution, without subsequent normalization to the same strength like in our scheme. We obtain $P \to N + 2$ and, thus, perfect

agreement if we adapt our setting such that the summed input strength $\sum_{j=1}^{N} \alpha_j^2$ fluctuates as it does in Ref. [9].

I. Input noise

Inputs in biological neural networks are noisy. To investigate the impact of input noise on WP and NP, we add white noise to all input neurons. Noise in the relevant inputs causes a finite error that remains even for optimal weights. Moreover, the irrelevant weights are not completely irrelevant anymore: They mediate noise (instead of zero) input, have an optimal value of zero, and lead to significant output error if they become too large. Since the input noise is different in the perturbed and unperturbed trials, it becomes a source of additional reward noise. We, thus, expect that larger input noise requires stronger weight or node perturbations to ensure that the beneficial, error gradient-related part of the reward signal is not dominated by reward noise (cf. also Ref. [33]). Furthermore, an increase in overall noise and additional weights that become more and more important should require the integration of more trials to extract gradient information. We, therefore, expect a reduction of the optimal learning rate with increasing input noise strength. Our numerical simulations confirm these points; see Fig. 5(a) (increase in task error and optimal error) and Supplemental Material Fig. S6 [28] (estimation of optimal learning rates and perturbation sizes).

We find that in the presence of input noise the irrelevant weights diffuse in WP and NP [Figs. 2(b)(iii) and 5(b)]. They settle at a finite spread, which contributes to the error [Fig. 5(c)]. The diffusion stops because WP and NP update irrelevant weights on average toward zero, due to their generation of errors. For WP, the final spread increases with decreasing noise strength and reaches infinity for zero noise. [Note that Fig. 5(b) does not show the final, stationary spread.] For NP, the diffusion of irrelevant weights is caused by their noise-induced updates and is in contrast to the noise-free case. Their final spread is independent of the noise strength and discontinuously drops to zero at zero noise. To explain this, we identify (weak) noise input with (weak) deterministic input and apply our findings for noise-free networks with inhomogeneous input strength distribution: For WP, each weight contributes equally to the final error, and the final spread scales like one over the square root of the input strength [Supplemental Material Sec. IV, Eq. (S126) [28]]. For NP, weights related to small inputs contribute only little to the error, and the final spread is independent of input strength [Supplemental Material Sec. IV, Eq. (S128) [28]].

The simulations [Fig. 5(a)] and our analytical understanding also show that for finite learning time or when introducing a weight-limiting mechanism there is no discontinuity in the error when increasing the input noise from zero to some finite value. The previous error analysis, therefore, stays valid as the limit of weak input noise. For WP, this is because limiting the learning time or the weights limits the final spread of irrelevant weights. This happens in such a way that sufficiently small noise has only a negligible effect on the output (Supplemental Material Sec. IV [28]). For NP and weak input noise, the final spread of irrelevant weights is approximately equal to that of the relevant weights and, thus, also limited.

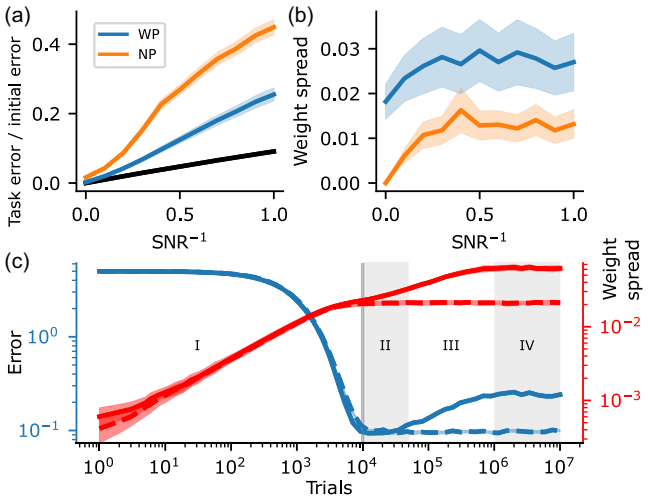

FIG. 5. Influence of input noise on the task error and on the spread of irrelevant weights. (a) Task error (fraction of the initial error) of WP (blue) and NP (orange) at 10 000 trials (curves, means; shading, standard deviations) as a function of inverse signal-to-noise ratio (SNR^{-1}). The noise in the relevant inputs renders the optimal error (black) nonzero. The plot covers SNRs ranging from infinity down to 1. (b) Spread (standard deviation) of irrelevant weights (mean and standard deviation) at 10 000 trials. (c) Evolution of error (blue, mean and SEM) and spread of irrelevant weights (red, mean and SEM) in WP proceeds for weak input noise ($\mathrm{SNR}^{-1} = 0.1$) in four phases (solid curve, phases indicated by numerals). Appropriate weight decay stops the dynamics in phase II and induces long-term results as in the noise-free case [dashed curve; cf. also Fig. 2(b)(ii)]. (a) and (b) use the learning rates that minimize error after 10000 trials [gray vertical line in (c)] for a given noise level (Supplemental Material, Fig. S6 [28]). This implies that WP's error in (a) is the one in phase II. Parameters: $M = N_{\mathrm{eff}} = 10$, $N = T = 100$, $\alpha^2 = N/N_{\mathrm{eff}} = 10$, $\sigma_{\mathrm{eff}} = 0.04$, $E_{\mathrm{opt}} = 0$, and $\gamma_{\mathrm{WD}} = 0.999\,98$.

Finally, we observe that WP learning proceeds for weak input noise in four phases [Fig. 5(c)]. In the first phase, the relevant weights are learned, such that the error decreases. Since the noise is small, the error due to its presence in relevant inputs is small. In the second phase, the error remains approximately constant, at a low level. In the third phase, the irrelevant weights, which have been diffusing all the time, become so large that they amplify the input noise sufficiently to influence the output despite the small noise amplitude. The error therefore increases. In the fourth and final phase, this error becomes so large that WP counteracts the further diffusion of weights. The error

therefore saturates, at a high level. The four phases can be clearly temporally separated. [Note the logarithmic axis scale chosen in Fig. 5(c).] We note that we observe divergence of the error for sufficiently large irrelevant weights if the learning rate is too large or the perturbations are too weak (Supplemental Material, Fig. S6 [28]). If learning stops in the second phase, the contributions from irrelevant weights can be neglected. The same holds if a limiting mechanism stops the diffusion of irrelevant weights at the level that they reach in the second phase, while only mildly affecting the relevant weights, because they converge at a shorter timescale [Fig. 5(c), dashed line: network with weight decay; cf. also Fig. 2(b)(ii)]. Interestingly, WP can then reach a lower (final) error than NP and is less affected by input noise; cf. Fig. 5(a).

J. Conclusions from the theoretical analysis and new learning rules

Our theoretical analysis reveals a simple reason for the differences between WP and NP: WP produces better perturbations, while NP better solves the credit assignment problem. Output perturbations caused by WP lie, in contrast to NP, always in the realizable output subspace and do not interfere with unrealizable target components. On the other hand, NP updates only (trial-)relevant weights, while WP updates all weights such that the (trial -)irrelevant weights change randomly. Small input noise does not change the overall picture. When single trials capture only a small part of the full task, WP learning slows down. Training in batches reduces the disadvantage.

Based on these insights, we introduce two novel learning rules, WP0 and hybrid perturbation (HP) [Figs. 6(a) and 6(b)]. WP0 adds a simple modification to WP: not to update currently irrelevant weights, i.e., weights whose inputs are zero (or close to it). This solves part of WP's credit assignment problem, as changing the weights does not improve the output, and it avoids diffusion of irrelevant weights. The improvement is especially large when inputs are sparse such that many inputs are (close to) zero [Figs. 6(a) and 6(c)], which might be frequently the case in biological neural networks [34–36]. HP aims to combine the advantages of WP and NP by generating the output perturbations like WP, through perturbing the weights, and generating updates like NP, using its eligibility trace. The learning rule performs well when all latent inputs have (approximately) the same strength α^2 [Figs. 6(b) and 6(c)].

WP0 and HP perform for the tasks used in the theoretical analysis section as well as WP and NP or better than both [Fig. 6(c)]. WP0 is, however, benefited by the assumption of rotated inputs (in contrast to WP and NP), as it renders the input maximally sparse. Furthermore, the latent inputs have equal strengths, benefiting HP. We observe only slight improvements of WP0 over WP for the reservoir computing and MNIST (Modified National Institute of Standards and Technology database) task, due to the lack of coding sparseness in our networks. HP performs much worse than WP and NP in the reservoir computing and similar to NP in the MNIST task. We explain this by the relevance of weak inputs [Supplemental Material Sec. VI, Eq. (S147) [28]]. Adding appropriately equalizing preprocessing layers may mitigate HP's problems. Furthermore, weak inputs may be irrelevant for biological learning. Since HP generates perturbations of the same class as the inputs and suppresses the learning of weights related to small inputs, we expect it to also work well for correlated input (as in Fig. 4) and in the presence of input noise.

K. Simulated learning experiments

In the following, we apply WP and NP to more general networks and temporally extended tasks with nonlinearities. We cover reservoir computing for dynamical

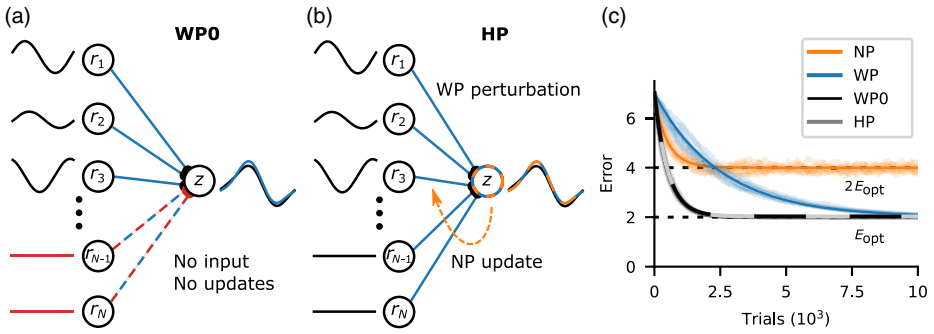

FIG. 6. New learning rules and learning of tasks consisting of multiple subtasks. (a) WP0 does not update weights that mediate zero input, avoiding their diffusion. (b) Hybrid perturbation (HP): NP scheme with output perturbations induced by WP. (c) WP converges approximately $P = 5$ (number of subtasks) times slower than NP, but in the presence of unrealizable target components [or for finite σ_{eff} and $N_{\text{eff}}^{\text{trial}} < T$; Supplemental Material Sec. III, Eqs. (S87)–(S89) [28]] to a lower final error. For the used maximally sparse and equally strong inputs, WP0 and HP combine the higher convergence rate of NP with the low final error of WP. Error curves (solid, theoretical predictions; shaded, ten exemplary runs) are for $M = 10$, $N, T = 100$, $N_{\text{eff}}^{\text{task}} = 50$, $N_{\text{eff}}^{\text{trial}} = 10$, negligible σ_{eff}, and $E_{\text{opt}} = 2$.

pattern generation, learning of recurrent weights in a delayed non-match-to-sample task, and a temporally extended, reward-based learning version of MNIST. The results confirm and extend our findings for analytically tractable tasks: They often show similar or superior performance of WP in temporally extended tasks relevant for biology and machine learning.

L. Reservoir computing-based drawing task

In reservoir computing schemes, an often low-dimensional input is given to a recurrent nonlinear network. The network effectively acts as a nonlinear filter bench: It expands the input and its recent past by applying a set of nonlinear functions to them. Each unit outputs one such function, which depends on the interactions within the recurrent network. Like a "computational reservoir," the network thereby provides in its current state the results of manifold nonlinear computations on the current and past inputs. A desired result can be extracted by training a simple, often linear readout of the reservoir neurons' activities. Reservoir computing schemes are widely used as models of neurobiological computations [37–41], since learning in them is simpler and seems more easily achievable with biological machinery than learning of full recurrent and multilayer networks. Furthermore, the schemes explain the presence of inhomogeneity and apparent randomness in neuron properties and connectivity in biological neural networks as helpful for enriching the computational reservoir. Here, we find that, when learning temporally extended output patterns with a reservoir computing scheme, WP can learn as well as or better than NP and NPc.

We consider a recurrently connected reservoir of $N = 500$ rate neurons driven by five external inputs of length $T = 500$. Inspired by the behaviorally relevant task of reproducing a movement from memory—here, drawing a figure—the task is to generate the x and y coordinates of a butterfly trajectory [42,43] at the $M = 2$ outputs by training a linear readout [Fig. 7(a)]. The trajectory is nontrivial in that it is not realizable from the external inputs. In fact, it requires reading out from many reservoir modes [Fig. 7(b), dashed gray line].

Formally, the task is similar to the setting discussed above, with the difference that there is a wide distribution of different, nonzero input strengths α_μ^2. The evolution of expected error is then best described by splitting the error $E = \sum_{\mu=1}^{N} E^\mu$ into different error components, each of which is associated with the weights that read out from a latent input $r_{\mu t}$ [Supplemental Material Sec. IV, Eq. (S103) [28]]. In WP and NP, the evolution of the error components follows a matrix exponential where different components decay at different rates and interfere with each other. Components that decay relatively quickly may be the main source of improvements in the beginning of training, whereas more slowly decaying components dominate the error toward the end. This effect can be seen in the approximately piecewise linear error decay in the logarithmic plot in Fig. 7(c).

Figure 7(c) compares the performance of WP, NP, and NPc in the drawing task. Perturbation size is finite, $\sigma_{\text{eff}} = 5 \times 10^{-3}$. WP converges faster initially, which may be typical for tasks with distributed input strengths [Supplemental Material Sec. IV, Eq. (S117) [28]]. It also achieves a lower final error. This is compatible with the observation that the effective dimension of the reservoir dynamics, as measured by the participation ratio (PR ≈ 5), is much smaller than the temporal extent of the task: The resulting smaller effective perturbation dimension of WP (MPR versus MT for NP versus $MT_{\text{eff}}^{\text{pert}}$ for NPc) yields an advantage for WP [Figs. 1(b) and 4(a)]. NPc reaches a lower final error than NP. We optimize its $T_{\text{eff}}^{\text{pert}}$ using a parameter scan (Supplemental Material, Fig. S8 [28]). From our simulations, we cannot completely exclude that

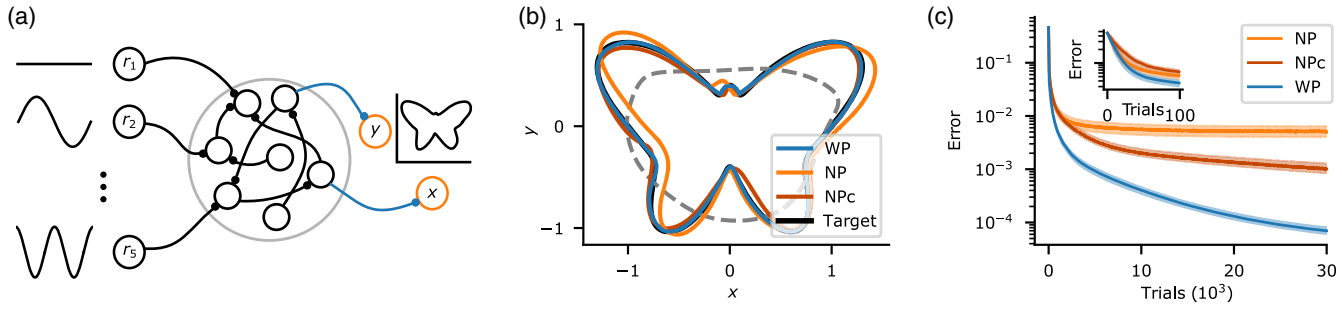

FIG. 7. WP outperforms NP on a reservoir computing-based drawing task. (a) Schematic of the recurrent, fixed reservoir receiving five external inputs. Only readout weights are learned. (b) Target (black) and final outputs of WP (blue), NP (orange), and NPc (red). A least squares fit (gray, dashed line) using only the first five principle components of the reservoir dynamics demonstrates that the task critically depends on reading out further, weaker dynamical components. (c) Error curves on a logarithmic scale. WP reaches a lower final error than NP and NPc, with NPc improving on NP; cf. also (b). Inset: early error evolution. There is a considerable improvement already during the first 50 trials. The curves show median (solid) and interquartile range between first and third quartile (shaded) over 1000 runs of the same network with different noise configurations.

the improvement is (in part) due to an effective decrease in learning rate, resulting from a longer correlation time in the perturbations than in some relevant inputs. For the biologically less relevant case of infinitesimal perturbation sizes, the performances of WP and NP are similar (Supplemental Material, Fig. S7 [28]) (compatible with Fig. 1 with $b = 0$). Toward larger trial numbers, the convergence of WP becomes slower: WP has difficulties with adjusting weights mediating weak inputs, since the impact of their perturbation on the error is small; the same effect underlies the weight diffusion in Fig. 2. Simulations indicate that the convergence is slower only by a constant factor of the order of 1 and that the optimal learning rate can be well estimated from the participation ratio (Supplemental Material, Fig. S9 [28]).

M. Delayed non-match-to-sample task

To ensure analytical tractability and for simplicity, so far we made a few biologically implausible assumptions. Specifically, only connection weights to linear units were trained, each trial consisted of a perturbed and an unperturbed run, and mostly the exact same input was used in each trial. In the following, we show that our findings generalize to settings without these assumptions. For this, we consider the learning of a DNMS task (temporal XOR) by nonlinear recurrent networks. DNMS tasks and closely related variants are widely used in both experiment [44] and theory [14,45], where they serve as simple working memory-reliant, not linearly separable decision-making tasks. We use the same setting as Ref. [14], which shows that a new variant of NP is able to solve the DNMS task. In particular, the setting is not adjusted to WP. The network consists of 200 nonlinear rate neurons receiving input from two external units u_1 and u_2. One of the network neurons, whose rate we denote with z, serves as its output [Fig. 8(a)]. In each trial, the network receives two input pulses, where each pulse is a 200-ms-long period with either u_1 or u_2 set to 1, and subsequently has to output 1 for 200 ms if different inputs are presented and -1 if the same inputs are presented [Fig. 8(b)]. There is a 200-ms-long delay period after each input pulse.

We train all recurrent weights using the usual update rules [Eqs. (3) and (6)] but replace the error of the unperturbed trial by an exponential average of the errors of the previous trials [12–14]. Hence, each trial now consists only of a perturbed and not additionally an unperturbed run. We first assume that the exact perturbations ξ are accessible for the weight update, which seems biologically plausible for WP (cf. Sec. III) but less so for NP (cf. Sec. III and Ref. [14]). Therefore, we also compare WP and NP to the biologically plausible version of NP proposed by Ref. [14], which avoids this assumption: In the weight update rule, it approximates the exact node perturbations ξ^{NP} with a nonlinearly modulated difference between the momentary input to a neuron and its short term temporal average (see Sec. IV for more details).

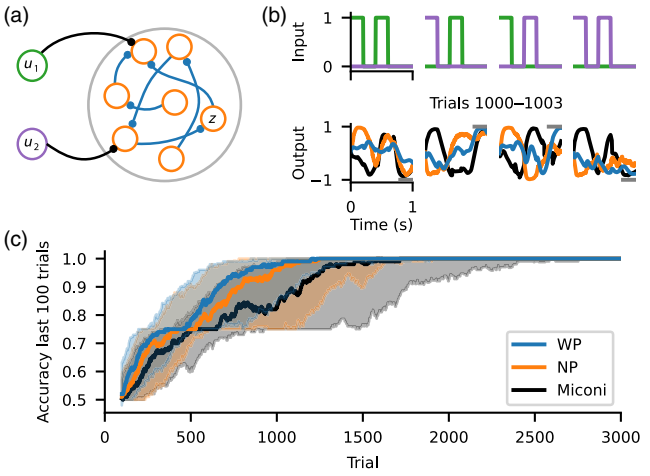

FIG. 8. WP performs as well as NP on a DNMS task. (a) Schematic of the recurrent network with inputs u_1 and u_2 and output z. All network weights are learned; i.e., for WP, all network weights (blue) are perturbed, and for NP, all network nodes (orange) are perturbed. (b) Inputs and outputs during example trials. Top row: inputs u_1 (green) and u_2 (purple) for the four different trial types. Bottom row: outputs for WP (blue), NP (orange), and the version of NP proposed by Ref. [14] (black) for trials 1000–1003 for the inputs shown above. Gray bars show target outputs. (c) Accuracy during training. WP (blue) performs similarly well as NP (orange) and the version of NP used by Ref. [14] (black). There is a noticeable transient slowdown at an accuracy of 75%, which corresponds to the successful learning of three out of the four different trial types. Solid lines show the median, and shaded areas represent the interquartile range between the first and third quartile using 100 network instances.

Figure 8(c) shows the performance of the three update rules in terms of their accuracy over the last 100 trials, where a trial is considered successful if the mean absolute difference between z and the target output is smaller than 1. We find that all update rules learn the task comparably well and reach perfect accuracy within at most 2000 trials when considering the median of network instances. Thus, our previous findings that WP can perform as well as or better than NP in simplified settings extend to the considered biologically plausible setup. That means WP can perform well for nonlinear neuron models, recurrent connectivity, and when the error of the unperturbed network is not available. Furthermore, the results indicate that approximating the perturbation as in Ref. [14] only mildly impacts the performance of NP for the considered task. Finally, we apply NPc to the task, which does not yield an improvement over NP (Supplemental Material, Fig. S12 [28]). Together with the similar performance of WP and NP, this indicates that the temporal dimension of the perturbation has little effect on task performance, perhaps because the period in which the target value needs to be assumed is rather short and the output is otherwise unconstrained.

N. MNIST

Finally, we apply WP and NP to MNIST classification. We use batches of images to train the networks. Each time step thereby corresponds to the presentation of one image, and the networks receive error feedback only at the end of a batch. This allows us to test how well WP and NP work on a more complicated, temporally extended task and in networks with a multilayer structure. In addition, it allows us to study how our analytical results for the learning of multiple input patterns (Sec. II G) extend to real-world tasks.

We use a two-layer feed-forward network with ten output neurons and 100 neurons in the hidden layer [Fig. 9(a)]. It learns via the rules [Eqs. (3) and (6)], where T equals the batch size N_{batch}. Hence, the perturbation is different for each image in the case of NP, while it is the same for WP. We test WP and NP for batch sizes of $N_{\text{batch}} \in \{1, 10, 100, 1000\}$. For each batch size, we determine the best-performing learning rates η and perturbation strengths σ^2_{WP} and σ^2_{NP} via grid searches. The perturbation strength has, however, little impact on performance, indicating that the final error is not restricted by reward noise due to finite size perturbations [Eqs. (13) and (14)].

We find that for WP the performance improves drastically with increasing batch size [Fig. 9(b)]. The final test accuracy is only about 69% for a batch size of 1 but reaches 92% for $N_{\text{batch}} = 1000$. Simultaneously, the optimal learning rate increases strongly, by a factor of approximately 50 [Supplemental Material, Fig. S10(c) and Table S2 [28]]. For comparison, the stochastic gradient descent (SGD) rule, which implements supervised not reward-based learning, reaches accuracies of 95%–98% for the considered batch sizes. In contrast, the learning curves of NP appear to be entirely independent of the batch size [Fig. 9(b)]; the final test accuracy is always about 86% and the optimal learning rate is constant as well. We also apply NPc to the task. The inputs are temporally uncorrelated, because the elements of the batches are drawn randomly. Based on our previous observations for uncorrelated input [Fig. 4(c)], we therefore expect that NPc performs similar to or worse than NP. The numerical experiments confirm this: Performance deteriorates with increasing perturbation correlation time; the effect is more pronounced with larger batch size (Supplemental Material Fig. S12 [28]). In conclusion, larger batch sizes, as commonly used in machine learning, favor WP, while smaller batch sizes favor NP (and NPc).

An improvement of WP with batch size and NP's independence of it are in agreement with our theoretical analysis Sec. II G. However, from this analysis we also expected that WP's learning rate can reach at most that of NP for large batch size. NP's slower convergence suggests that it is more susceptible to deviations of the network architecture from linear, single-layer networks. Indeed, when using single-layer networks, NP's performance improves, while the opposite holds for WP and SGD (Supplemental Material Fig. S11 [28]). In a single-layer linear network with realizable targets, NP performs better than WP even for large batch sizes (Supplemental Material Fig. S11 [28]), consistent with our analytical findings that training with different subtasks (here, different batches) harms WP (Sec. II G) while the absence of unrealizable targets benefits NP (Sec. II E).

The results are particularly remarkable when naively comparing the number of perturbed nodes and weights: For the network considered here, there are only 110 output and hidden nodes but 79 510 weights (including biases). Nevertheless, WP can clearly outperform NP. Also, a

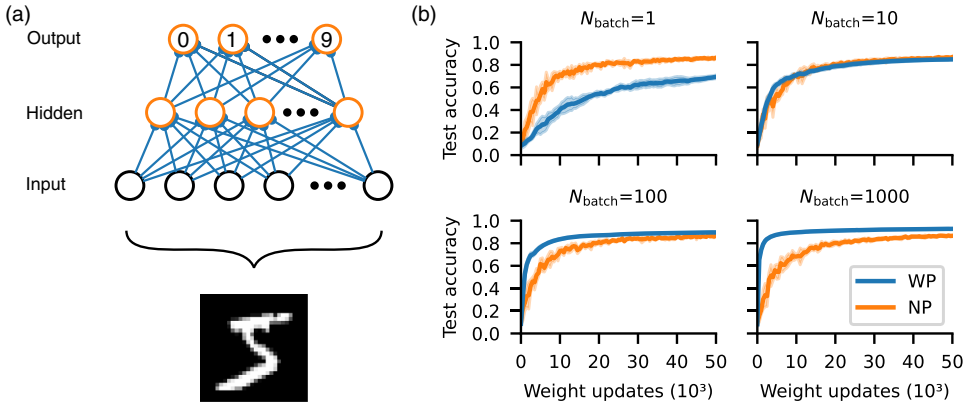

FIG. 9. WP can outperform NP on MNIST. (a) Schematic of the used fully connected, two-layer network. All network weights are learned; i.e., for WP all network weights (blue) are perturbed, and for NP all network nodes (orange) are perturbed. (b) Test accuracy as a function of the number of weight updates for WP (blue) and NP (orange) for different batch sizes. NP does not profit from increasing the batch size and always reaches a final accuracy of approximately 86%. WP improves considerably with increasing batch sizes and reaches a final accuracy of approximately 92% for $N_{\text{batch}} = 1000$. Solid lines show the mean, and shaded areas show the standard deviation using five network instances.

comparison of the actual perturbation dimensions cannot explain the better performance of WP in, e.g., Fig. 9(b) lower left (WP pert. dim., 79 510; NP pert. dim., $110 \times T = 11\,000$).

III. DISCUSSION

Our results show that WP performs better than NP for tasks where long trials capture most of the task's content. This might seem paradoxical, as NP incorporates more structural knowledge on the network, namely, the linear summation of inputs. However, WP accounts for the fact that the weights in a neural network are (approximately) static. Furthermore, by perturbing the weights, it implicitly accounts for low input dimensionality and generates only realizable output changes. Therefore, it generates better tentative perturbations. This leads to less noise in the reward signal and better performance (smaller final error and sometimes faster convergence) in the tasks where WP is superior to NP.

Our theoretical analysis shows that the lower noise in WP first results from an effective perturbation dimension that is lower than NP's if the temporal extent of a task is larger than its input dimensionality, $T > N_{\mathrm{eff}}$. Second, factors such as the attempt of NP to realize unrealizable targets contribute. Temporally extended tasks with durations on the order of seconds and low dimensionality occur frequently in biology, for example, in motor learning and working memory tasks. In line with perturbation-based learning, biological movements are endowed with noise, which helps their learning and refinement [46]. The associated neuronal dynamics in the brain are confined to a low-dimensional space, a property shared by many types of biological and artificial neural network activity [47–49]. The dynamics for simple movements as investigated in typical experiments are embedded in spaces of dimension of the order of 10 [19]. This indicates low effective input dimensionality N_{eff} at the different processing stages. The effective muscle activation dimensionality is similarly low [20,50]. Neurons under *in vivo* conditions can faithfully follow input fluctuations on a timescale of 10 ms [51], and significant changes in neuronal trajectories happen on a timescale of 100 ms [19,21,52]. For the learning of a movement of duration 1 s, this suggests an effective temporal dimension of about 10–100 similar to the expected input dimension. This implies that WP as well as NP and NPc are promising candidates for the learning of simple movements. Our results indicate that WP is superior if the movements are longer lasting or lower dimensional.

We explicitly study the learning of movement generation (drawing task) and of a working memory task (DNMS). The numerical simulations show that WP performs similarly well or better compared to NP. In a task generally investigating the learning of complicated nonlinear, temporally extended input-output tasks (MNIST), WP outperforms NP as soon as the tasks have sufficient temporal extent.

As another concrete application, consider the learning of the single song in certain birds. A single, stereotypical input sequence in a "conductor area" (HVC) may drive the circuit [35,53]. The effective input dimension N_{eff} is, thus, at most as large as the temporal dimension T of the task. Based on recent experiments, Ref. [53] proposes that the output of the tutor and experimenter area (LMAN) is modified by reinforcement learning via NP, such that it guides the motor area (RA) to learn the right dynamics. Our analytical results predict that WP is as well or better suited to achieve this task, since $N_{\mathrm{eff}} \leq T$. Earlier work suggests that WP [22] or NP [17] may directly mediate the learning of the connections from HVC to RA. Because of HVC's very sparse activity, WP0 is highly suitable for such learning. Reward-based learning of mappings between conductor sequences and downstream neural networks may also be important for different kinds of precisely timed motor activity [54,55] and for sequential memory [56,57].

Biological neural networks are inherently noisy. We find that WP and NP induce two types of weight update noise: credit assignment and reward noise. We understand their impact analytically. Additional feedback or output noise implies additional reward noise with like effects. We, thus, additionally study only the impact of input noise on the learning of linear networks. Our simulations show that the convergence time and the final error increase with the input noise strength. The increase is smaller in WP than in NP. We further find that our results with zero noise are recovered in the limit of small noise compared to the strength of the relevant latent inputs, if the learning time is finite. The same holds for WP also if the weights are appropriately limited, for example, due to weight decay. The fact that animals can learn to perform tasks with high precision, i.e., with small final error, indicates that the case of small noise may be the biologically relevant one. The results of WP and NP learning with noisy inputs can be understood analytically from our findings on inhomogeneous input distributions (Supplemental Material Sec. IV [28]). Also, the considered DNMS and MNIST tasks contain input noise: the DNMS task because of the randomly chosen initial conditions and the MNIST task because of the naturally noise-afflicted input images. We conclude that the finding of similar or better performance of WP compared to NP in temporally extended, low-dimensional tasks may readily apply to biologically plausible, noisy networks.

WP and NP have biologically plausible implementations. NP requires that the plastic synapses can keep track of their input and the somatic perturbations (which may arrive from a tutor or experimenter neuron). Biologically plausible mechanisms for this have been proposed for tasks with both immediate reward [12,13] and reward at a temporally extended trial's end [14]. Their underlying idea is to assume that the perturbation fluctuates more quickly than the other input. The present fluctuation can then be approximately isolated by subtracting a short-term

temporal average of the past overall input from the present one [12,13]. This difference replaces the injected perturbation in the eligibility trace. For tasks with late reward, the eligibility trace needs to integrate a nonlinearly modulated version of the described difference [14]. This prevents the cancellation of a perturbation's effect by the subsequent change in the average that it evokes, because the peak in the original perturbation is sharper and higher than the one in the average. We use this learning model of Ref. [14] in Fig. 8. The biological implementation of WP may be even simpler. A neural network needs to generate labile random weight changes and keep track of them. They should be approximately constant during a task and enhanced, deleted, or reversed by a subsequent reward signal. Experiments on timescales from minutes to days find spontaneous changes in the synaptic weights, which have similar strength as changes due to activity-dependent plasticity [31]. Such changes might generate the perturbations required for our WP scheme. Previous work suggests also synaptic unreliability to provide the perturbations for WP [58]. This fits into our scheme of static weight perturbations if neurons spike once during a trial or if they burst once and the synaptic transmission is restricted to a single time bin. Another source of the required randomness may be fluctuations of activity-dependent plasticity, while the deterministic baseline acts as a useful prior. If the baseline is unrelated to the task, it is with high probability orthogonal to task-relevant directions (due to the high-dimensional weight space) and not harm learning, similar to the weight diffusion in WP. In this way, the fluctuations of activity-dependent plasticity, rather than their deterministic part, may be the source of learning.

Modulation of weight changes by reward is observed in various experiments [59,60]. As an example, the potentiation of synapses is enhanced or reversed depending on the presence or absence of a temporally close dopamine reward signal [61]. Also, other factors play a role; potentiation can, for example, be reversed within a "grace period" of tens of minutes by a change of environment [62]. The consolidation and amplification of changes may be dependent on plasticity-related proteins, which are upregulated by reward and for which the synapses compete (synaptic tagging hypothesis) [60,63]. *A posteriori* modifications of tentative synaptic weight changes are also assumed in the reinforcement learning scheme of reward-modulated Hebbian plasticity [64,65], which is closely related to WP.

WP applies a random perturbation vector to the weights, measures the error change to obtain a reinforcement signal, and applies as weight update the perturbation modulated by the reinforcement signal. The improvement, therefore, follows on average the weight gradient. A related approach is to randomly perturb and accept the perturbation if it leads to a better performance [22]. This simple instance of an evolutionary strategy [66,67] is also applicable if there is no gradient. Our results for WP suggest that this and related evolutionary learning strategies might benefit from tasks that are low dimensional, as reported previously [68], and not be harmed by their temporal extension. The sketched simple evolutionary learning may in the brain generate structural improvements: Experiments show that synaptic turnover occurs in the presence but also spontaneously in the absence of neuronal activity [69–72]. This may reflect the random elimination and creation of synapses and their consolidation by activity-dependent plasticity [73–78]. The basis of consolidation is that mainly weak synapses are removed such that strengthening through Hebbian learning causes the long-term presence of a synapse. Furthermore, Hebbian learning counteracts spontaneous synaptic weight changes, which could otherwise weaken useful synapses and ultimately lead to their removal. Network models show that restructuring with subsequent selective consolidation can recruit sparse available connectivity for task learning, prevent catastrophic forgetting, and may explain the benefits of dividing learning into several temporally distinct phases [73,75,77]. Furthermore, it may explain the experimental observation that there are commonly multiple synapses between connected neurons [74,76,78]. The signal for the strengthening of a tentatively established synapse may be interpreted as a reinforcement signal for its presence. This signal is generated if the pre- and postsynaptic neurons are coactive, due to external stimulation, or recall mediated by other, already strengthened synapses. In contrast to WP, the reinforcement signal is, therefore, specific to a neuronal connection (consisting of the possible multiple direct synapses from a presynaptic to a postsynaptic neuron), which simplifies learning. In particular, any useful new synapse is consolidated (unless the coactivity of the pair of connected neurons stops due to other changes). In contrast, in WP, tentatively applied useful weight changes can be easily reverted due to harmful ones in other parts of the weight perturbation vector. In a related model, Ref. [79], consolidatory strengthening is implemented with NP instead of Hebbian learning. This allows one to learn tasks based on a global reward signal. The synaptic weight fluctuations in the model induce random changes in task-irrelevant directions similar to the weight diffusion that we observe in WP. Our results suggest to directly exploit the synaptic weight fluctuations for consolidatory synaptic strengthening by using WP when learning low-dimensional and temporally extended reward-based tasks.

WP is proposed in several variants. They differ in (i) the task setup, for example, instantaneous [7,9] or temporally extended tasks [8,22,58], (ii) the implementing network, for example, rate [22] or spike-based [58,80] networks, (iii) the perturbation scheme, where all weights

are simultaneously perturbed [7,8,58] or only one weight at a time [15], (iv) the computation of the weight update, by correlating reward and perturbation [7–9,11,58] or direct estimation of the gradient components (for the single weight perturbation scheme) [9,15], (v) the estimation of the success of the perturbed network, which may involve a comparison of the obtained reward to an unperturbed baseline [8,9] or a running average [22,58] or it may consider the reward only [7,58], and (vi) the weight update, which may be proportional to the success of the network [7–9,11,58] or independent of its size as long as there is an improvement [22]. A similar diversity of NP variants exists [9–14,16,17,24,79,81,82].

The tasks considered in our article are temporally extended. The reward is provided at the end of the trial but influenced by earlier output states. This is consistent with many tasks in biology [1,5,14,22] and with the learning schemes by Refs. [8,10,14,22]. We choose a WP rule that is biologically plausible, as it involves simultaneous perturbations to all weights and correlates reward and weight change. The success measure compares the obtained reward to the reward of an unperturbed network in order to reduce the update noise [8,9]. In particular, this avoids unfavorable perturbations being associated with positive reward feedback. Finally, the weight update is proportional to the measured success in order to ensure that it occurs on average parallel to the reward gradient. The choices are identical to those by Refs. [8,9] for temporally not extended tasks. Specifically, the results in Ref. [9] appear as a special case of our results for multiple input patterns; if the task dimension is maximal, single trials have no temporal extent, and the inputs have fluctuating amplitude (see Sec. II H).

We choose the NP scheme such that it matches the WP scheme. It is a discrete-time version of the NP scheme proposed by Ref. [10] and an extension of the scheme by Ref. [9] to temporally extended tasks. In biologically plausible implementations of WP and NP, the reward should be compared to an intrinsically generated prediction, such as an average of previous rewards [12–14] or the reward of another perturbed trial [82]. In the delayed nonmatch-to-sample task, we thus replace our standard unperturbed baseline by such an average. This also allows a direct comparison with the NP scheme by Ref. [14]. In Sec. II I, the perturbed and unperturbed trials have different input noise, such that E is no longer the exact unperturbed counterpart of E^{pert}.

To exploit correlations in the inputs with a node perturbation learning rule, we introduce NPc, which is identical to standard NP apart from using temporally correlated, smoothed node perturbations. We find that the temporal correlations are usually beneficial if also the inputs are (similarly) correlated. This is in contrast to Ref. [13], which observes a detrimental effect already of short correlations for a node perturbation variant that relies on high-frequency perturbations. Other previous studies inject white noise perturbations only [10,12,16,17]. Our simulations with linear networks indicate that the perturbation correlation time of NPc should optimally match that of the inputs; NPc then performs similar to NP in a task with reduced temporal extension.

NP is studied in various concrete neurobiological settings. Previous work uses feedforward networks with NP to model the learning of coordinate transforms in the visual system [83], birdsong [17,53], and motor output [12,84]. Reference [13] shows that reservoir computers with NP trained, fed back readouts can learn periodic inputs, routing, and working memory tasks. Reference [14] uses a fully plastic recurrent network for the learning of a delayed non-match-to-sample, a selective integration, and a motor control task. Finally, NP is often employed for reference and comparison [85–91]. WP is considered less in studies of neurobiological learning. It is implemented in early feedforward network models of birdsong [92] and binary output task learning [58,80]. Furthermore, it is occasionally used for comparison [86,88]. Very recently, Ref. [4] has shown that recurrent neural networks can be pretrained with WP and the reservoir computing scheme to thereafter learn with static weights to generate fixed point activity.

The results of our present article using feedforward, reservoir computing and fully plastic recurrent networks suggest that for many tasks WP is at least as suitable as NP, while the implementation may be even simpler. This indicates that WP is a useful benchmark and a similarly plausible model for learning in the brain as NP. Experimentally measurable features of the learning and weight dynamics may allow one to distinguish the learning rules in biological neural networks.

IV. MATERIALS AND METHODS

A. Analytical error dynamics

To analytically compute the dynamics of the expected error, we consider an arbitrary perturbation ξ. This determines the error change $E^{\mathrm{pert}} - E$ and the resulting weight update Δw via Eqs. (3) and (6). Δw, in turn, determines the new weights and via Eq. (8) the error $E(n+1)$ after the update. $E(n+1)$ is, thus, a function of ξ, the weight mismatch $W(n)$ before the update, and the input correlations S:

$$E(n+1) = \frac{1}{2}\mathrm{tr}\{[W(n)+\Delta w(\xi)]S[W(n)+\Delta w(\xi)]^T\}. \quad (17)$$

Averaging over perturbations and using Isserlis' theorem yields an equation for the expected error $\langle E(n+1)\rangle$. When assuming that all latent inputs have the same strength, $\langle E(n+1)\rangle$ becomes a function of the error $\langle E(n)\rangle$ before the update and the system parameters, leading to Eq. (9). The detailed derivation is given in Appendix B.

B. Numerical simulations accompanying the theoretical analysis

In the numerical experiments in Sec. II B, the N_{eff} nonzero inputs are orthonormal functions, superpositions of sines, scaled by $\alpha^2 = N/N_{\text{eff}}$ to keep the total input strength $\alpha^2 N_{\text{eff}}$ for different N_{eff} constant. Targets z_{it} are obtained by linearly combining these functions using teacher weights $w^*_{ij} = 0.1$ and adding as an unrealizable component a further, appropriately scaled, orthonormal function. Learning rates are η^*.

C. Input and perturbation correlations

NPc is applicable to general network dynamics (cf. Fig. 7 and Figs. S7, S8, and S12 [28]). In Fig. 4, we apply it to linear networks with correlated inputs that are constructed similar to the perturbations, by low-pass filtering N_{eff} white noise traces (filtering time constant $\tau_{\text{corr}}^{\text{input}}$ and effective temporal dimension $T_{\text{eff}}^{\text{input}}$). Subsequently, we orthonormalize them, which somewhat modifies the correlation times. The realizable components of a target are linear combinations of these correlated inputs weighted by $w^*_{ij} = 0.1$. We assume that there is an additional unrealizable component, which, for simplicity, contains all modes orthogonal to the inputs with equal strength, such that $E_{\text{opt}} = 2$.

For $T_{\text{eff}}^{\text{pert}} \geq T_{\text{eff}}^{\text{input}}$, NPc operates at $\eta_{\text{NPc}} = \eta^*_{\text{NP}}$, the optimal learning rate for NP. For $T_{\text{eff}}^{\text{pert}} < N_{\text{eff}}$, we use the optimal learning rate of NP for a task with reduced $N_{\text{eff}} = T_{\text{eff}}^{\text{pert}}$, i.e., $\eta_{\text{NPc}} \approx (N_{\text{eff}}/T_{\text{eff}}^{\text{pert}}) \cdot \eta^*_{\text{NP}}$. Our intuition is that perturbations with temporal dimension $T_{\text{eff}}^{\text{pert}} < N_{\text{eff}}$ can only improve an $MT_{\text{eff}}^{\text{pert}}$-dimensional subspace of the weights. For NPc, $T_{\text{eff}}^{\text{pert}} < N_{\text{eff}}$ therefore reduces the learnable number of weights from MN_{eff} to effectively $MT_{\text{eff}}^{\text{pert}}$ independent ones. This is like in Fig. 1(c), gray curves, where T restricts N_{eff}. As effectively fewer weights are learned, a higher learning rate can be chosen. We perform additional simulations with an unadjusted (smaller) learning rate that confirms our choice, as convergence otherwise becomes much slower.

For a given $T_{\text{eff}}^{\text{pert}}$, simulations of 20 000 trials are repeated 1000 times (randomly generated inputs change between runs but not within a run). The final error of a run is computed by averaging over the last 500 trials (in which the error is approximately constant; Supplemental Material Fig. S4 [28]) to determine the mean over runs and its SEM. To determine $n_{\text{trials}}^{\text{decay}}$, we use ten samples of 100 runs each. For each sample, we compute the mean error over runs and additionally smooth it with a centered temporal running average of window size 20. $n_{\text{trials}}^{\text{decay}}$ is then the trial for which the described average drops for the first time below $E_{f,\text{unr}} + 0.05 \cdot [E(0) - E_{f,\text{unr}}]$. Figure 4(b) reports the mean and standard error of the mean of $n_{\text{trials}}^{\text{decay}}$ over all samples. Figure 4(c) repeats the same analysis for NP with η varied from $0.05 \cdot \eta^*_{\text{NP}}$ to η^*_{NP}. Gray curves in Fig. 4(c) (NPc with η adjusted by a factor of $0.5, 0.6, \ldots, 1.2$) use 100 repetitions.

D. Input noise

We extend the basic theory task, in which a single mapping from an effectively N_{eff}-dimensional input signal r_{jt} onto target outputs $z^*_{it} = \sum_{j=1}^{N} w^*_{ij} r_{jt}$ is learned, by adding independent white noise to each input at each time step:

$$r_{jt}^{\text{noisy}} = r_{jt} + \chi_{jt}, \qquad \langle \chi_{jt} \chi_{ks} \rangle = \sigma^2_{\text{noise}} \delta_{jk} \delta_{ts}. \quad (18)$$

As the added white noise has a rotationally symmetric distribution in the space of input neurons, we can still without loss of generality rotate the input space such that each of the first N_{eff} input neurons carries a signal component and additional noise, while the remaining inputs are purely noisy. We note that, because the noise in the task-relevant inputs is amplified by the weights, their optimal values are closer to zero than those of the noise-free task.

SNR is defined as the ratio of the total (summed) power in the input signal to that in the noise. Averages are taken over 100 repetitions [Figs. 5(a)–5(c)], the last 1000 of 100 000 trials [Figs. 5(a) and 5(b)], and over irrelevant weights [Fig. 5(c)].

E. Reservoir computing task

The $N = 500$ rate neurons of the fully connected recurrent reservoir network evolve according to

$$x_{jt} = \gamma x_{j,t-1} + (1-\gamma)\left(\sum_{k=1}^{N} w_{jk}^{\text{rec}} r_{k,t-1} + \sum_{q=1}^{N_{\text{inputs}}} w_{jq}^{\text{in}} r_{qt}^{\text{in}} \right). \quad (19)$$

The rate of neuron k is $r_{kt} = \tanh(x_{kt})$. Their decay time constant is $\tau = 10$ time steps, i.e., $\gamma = e^{-1/\tau}$. Recurrent weights w^{rec} are drawn from a centered normal distribution; the weight matrix is thereafter normalized to ensure that the real part of its largest eigenvalue is $g_{\text{rec}} = 1$. Input weights w^{in} are drawn from $w_{jk}^{\text{in}} \sim \mathcal{N}(0, 1/N_{\text{in}})$. Creating various instances of such random networks shows that performance and participation ratio $\text{PR} = (\sum_{\mu=1}^{N} \alpha_\mu^2)^2 / \sum_{\mu=1}^{N} \alpha_\mu^4$ are rather independent of the instance. The participation ratio gives an estimate of the dimensionality of the reservoir dynamics [19,93]. Generally, we observe $\text{PR} \approx 5$; for example, in the network in Fig. 7, $\text{PR} \approx 5.3$. The $N_{\text{in}} = 5$ inputs to the reservoir are orthogonal to each other, $r_{1t}^{\text{in}} = 1$, $r_{2t}^{\text{in}} = \sqrt{2}\sin(\omega t)$, $r_{3t}^{\text{in}} = \sqrt{2}\cos(\omega t)$, $r_{4t}^{\text{in}} = \sqrt{2}\sin(2\omega t)$, $r_{5t}^{\text{in}} = \sqrt{2}\cos(2\omega t)$, $\omega = 2\pi/T$, and $T = 500$ time steps. The trained linear readout produces $M = 2$ outputs $z_{it} = \sum_{j=1}^{N} w_{ij} r_{jt}$. Their target z_t^* is, up to scaling, the same

as in Ref. [42]: $z^*_{1t} = \text{radius}_t \cos(\omega t)$, $z^*_{2t} = \text{radius}_t \sin(\omega t)$, with $\text{radius}_t = 0.1 \cdot [9 - \sin(\omega t) + 2\sin(3\omega t) + 2\sin(5\omega t) - \sin(7\omega t) + 3\cos(2\omega t) - 2\cos(4\omega t)]$. Already 100 time steps before the task starts, the reservoir is initialized and given external input. By the time the task begins, network activity is enslaved by the external input and settles down to a periodic orbit. Technically, we record the reservoir activity traces r_{jt} once for the entire training of w, because they are the same in each trial. The value of the participation ratio motivates us to construct an optimal readout reading out the largest five principal components via the least squares fit $w^{\text{LS}}_{ij} = \sum_{k=1}^{N} \sum_{t=1}^{T} z^*_{it} r_{kt} S^{\text{pinv}}_{kj}$ [Fig. 7(b), dashed gray line]. Here, S^{pinv} is the pseudoinverse of the reduced correlation matrix of the reservoir that is obtained by setting all eigenvalues of S except the largest five to zero. Including six principal components does not qualitatively change the result.

From the theoretical analysis Eq. (12), we obtain an estimate $\eta^* = 1/[(M\text{PR} + 2)\overline{\alpha^2}]$ for the optimal learning rate, by setting $N_{\text{eff}} \to \text{PR}$ and $\alpha^2 \to \overline{\alpha^2} = (1/\text{PR}) \sum_{\mu=1}^{N} \alpha_\mu^2$. $\overline{\alpha^2}$ is the strength of each latent input when we assume that the total input strength is generated by PR equally strong ones. We verify by a grid search that this estimated value yields for both WP and NP close to optimal performance, as measured by the error after 10 000 trials with infinitesimally small σ^2_{eff}, showing that it indeed maximizes convergence speed. We, therefore, choose it as the learning rate for our task with infinitesimal (Supplemental Material Fig. S7 [28]) and also with finite perturbation size (Fig. 7), since the theoretical analysis yields independence of η^* from σ^2_{eff} [Eq. (12)]. For NPc, we use the same learning rate as for NP and WP and determine the optimal $T^{\text{pert,opt}}_{\text{eff}} = 18$ by minimizing the error after 30000 trials (Supplemental Material Fig. S8 [28]). Simulations with finite perturbations use $\sigma_{\text{eff}} = 5 \times 10^{-3}$. A scan over σ_{eff} confirms that the final error depends quadratically on it, as predicted by the theory.

F. Delayed non-match-to-sample task

The fully connected recurrent network has $N = 200$ rate neurons. The dynamics of neuron i, $i = 4, \ldots, N$, are governed by

$$\tau \dot{x}_i = -x_i(t) + \sum_{j=1}^{N} w^{\text{rec}}_{ij} r_j(t) + \sum_{q=1}^{2} w^{\text{in}}_{iq} u_q(t), \quad (20)$$

with time constant $\tau = 30$ ms. The constant activations $x_1(t) = x_2(t) = 1$ and $x_3(t) = -1$ provide biases [14]. The rate of each neuron i, $i = 1, \ldots, N$, is given by $r_i(t) = \tanh[x_i(t)]$. $z(t) = r_4(t)$ is the network output. We use the forward Euler-method with step size $dt = 1$ ms to simulate the dynamics and draw the initial activations from a uniform distribution, $x_i(0) \sim \mathcal{U}(-0.1, 0.1)$ for $i = 4, \ldots, N$. Recurrent weights are drawn from a Gaussian distribution, $w^{\text{rec}}_{ij} \sim \mathcal{N}(0, g^2/N)$, with $g = 1.5$. Input weights are drawn from a uniform distribution, $w^{\text{in}}_{iq} \sim \mathcal{U}(-1, 1)$.

All recurrent weights w^{rec}_{ij} are trained. The error function of WP and NP is the mean squared difference between the output z and the target within the last 200 ms of each trial. For each of the different trial types k, $k = 1, \ldots, 4$, we use an exponential average of the previous errors $E^{\text{pert}}(n_k)$ for this trial type (n_k indexes the trials of type k) as the error baseline:

$$E_k(n_k) = E_k(n_k - 1) + \frac{1}{\tau_E}[E^{\text{pert}}(n_k) - E_k(n_k - 1)], \quad (21)$$

where $\tau_E = 4$. To get the best-performing learning parameters, we perform a grid search, which yields $\eta^{\text{WP}} = 1 \times 10^{-5}$, $\sigma^{\text{WP}} = 4.64 \times 10^{-3}$, $\eta^{\text{NP}} = 1 \times 10^{-5}$, and $\sigma^{\text{NP}} = 4.64 \times 10^{-1}$.

For the details of the version of NP proposed by Ref. [14], see this article. For the convenience of the reader, here we briefly mention the main differences to the vanilla NP version Eq. (6): For each network neuron, a node perturbation is applied at a simulation time step only with a probability of 0.3% and is drawn from a uniform distribution, $\xi \sim \mathcal{U}(-16, 16)$. The error is given by the absolute difference between output and target. Weight updates are computed via $\Delta w^{\text{rec}}_{ij}(n_k) = -\eta E_k(n_k - 1)[E^{\text{pert}}(n_k) - E_k(n_k - 1)] \sum_{t=1}^{T} [(x_{it} - \bar{x}_{it}) r_{j,t-1}]^3$ and clipped when they exceed $\pm 3 \times 10^{-4}$ (cf. code accompanying Ref. [14]). t indexes the simulation time step of each trial, T is the total number of simulation time steps per trial, and $\bar{x}_{it} = \bar{x}_{i,t-1} + (1/\tau_x)(x_{it} - \bar{x}_{i,t-1})$ is an exponential average of past activations. Parameter values are $\eta = 0.1$ and $\tau_x = \frac{20}{19}$.

G. MNIST classification task

The input layer of the fully connected feedforward network consists of 784 units encoding the pixel values of the data. The hidden layer consists of 100 neurons with tanh activation function and biases. The output layer consists of ten neurons, one for each single-digit number, with softmax activation function and biases. We use the standard training and test dataset but split the standard training data set into a training dataset of 50000 images and a validation dataset of 10 000 images. No preprocessing is done on the data. We employ vanilla WP Eq. (3), NP Eq. (6), or SGD to train all parameters of the network. The error function is the cross-entropy loss averaged over the batch of length $N_{\text{batch}} = T$. We also try to combine the gradient estimates obtained from WP and NP with Momentum, RMSProp, or Adam [29] but do not find an improvement of performance compared to the vanilla versions with carefully tuned parameters. The same holds for SGD. This may be because of the rather simple network architecture.

To obtain the best-performing parameters (the learning rate for all three algorithms and the standard deviation for WP

and NP), we perform a grid search for each of the considered batch sizes: For each parameter set, we train the network for 50 000 trials (i.e., weight updates) on the training dataset. We then select the best-performing parameter sets based on the final accuracy on the validation dataset and apply them to the test dataset. High final accuracy appears to concur with fast convergence speed, such that a comparison to our analytical results (where learning rate optimizes the convergence speed) seems justified.

The supporting data for this article are openly available from GitHub [94].

ACKNOWLEDGMENTS

We thank the German Federal Ministry of Education and Research (BMBF) for support via the Bernstein Network (Bernstein Award 2014, 01GQ1710).

APPENDIX A: LEARNING MODELS AND TASK PRINCIPLES

1. Mean updates for small perturbations

To calculate the average update of WP, one uses the linear order approximation valid for small perturbations [7,8]:

$$E^{\mathrm{pert}} - E \approx \sum_{m=1}^{M}\sum_{k=1}^{N} \frac{\partial E}{\partial w_{mk}} \xi_{mk}^{\mathrm{WP}}, \tag{A1}$$

where the sum is over all weights in the network. With this and $\langle \xi_{ij}^{\mathrm{WP}} \xi_{mk}^{\mathrm{WP}} \rangle = \delta_{im}\delta_{jk}\sigma_{\mathrm{WP}}^2$, where δ is the Kronecker delta, the WP update rule [Eq. (3)] yields

$$\langle \Delta w_{ij}^{\mathrm{WP}} \rangle \approx -\frac{\eta}{\sigma_{\mathrm{WP}}^2} \sum_{m=1}^{M}\sum_{k=1}^{N} \frac{\partial E}{\partial w_{mk}} \langle \xi_{mk}^{\mathrm{WP}} \xi_{ij}^{\mathrm{WP}} \rangle = -\eta \frac{\partial E}{\partial w_{ij}}; \tag{A2}$$

i.e., the weight update is along the negative error gradient. The result's independence of σ_{WP} motivates the division by σ_{WP}^2 in Eq. (3).

NP perturbs for temporally extended tasks the total inputs $y_{it} = \sum_{k=1}^{N} w_{ik} r_{kt}$ of neurons $i = 1, \ldots, M$ by the node perturbation vectors ξ_{it}^{NP}. Therefore, the change in the scalar error is, to linear order, given by the projection of ξ_{it}^{NP} onto the error gradient $\partial E/\partial y_{it}$ with respect to y_{it} [10]:

$$E^{\mathrm{pert}} - E \approx \sum_{i=1}^{M}\sum_{t=1}^{T} \frac{\partial E}{\partial y_{it}} \xi_{it}^{\mathrm{NP}}. \tag{A3}$$

Since $\partial y_{it}/\partial w_{ij} = r_{jt}$, the gradients with respect to weights and sums of inputs are related via the chain rule by $\partial E/\partial w_{ij} = \sum_t \partial E/\partial y_{it} r_{jt}$. This reveals that the NP weight update [Eq. (6)] is on average along the negative error gradient:

$$\begin{aligned}\langle \Delta w_{ij}^{\mathrm{NP}} \rangle &\approx -\frac{\eta}{\sigma_{\mathrm{NP}}^2} \sum_{m=1}^{M}\sum_{s,t=1}^{T} \frac{\partial E}{\partial y_{mt}} \langle \xi_{mt}^{\mathrm{NP}} \xi_{is}^{\mathrm{NP}} \rangle r_{js} = -\eta \sum_{t=1}^{T} \frac{\partial E}{\partial y_{it}} r_{jt} \\ &= -\eta \frac{\partial E}{\partial w_{ij}}.\end{aligned} \tag{A4}$$

Equations (A2) and (A4) hold for any error function. For corresponding more specific computations for the quadratic error function [Eq. (8)], see Appendix A3.

2. Dependence of weight update noise on error baseline

To show that the choice of E as error baseline in Eq. (3) minimizes the update noise for WP, we compute the variance $\langle\langle \Delta w_{ij}^{\mathrm{WP}} \rangle\rangle$ of the WP weight updates when adding a possibly trial- and synapse-dependent baseline term $\tilde{E}_{ij}$ to the update rule, $\Delta w_{ij}^{\mathrm{WP}} = -(\eta/\sigma_{\mathrm{WP}}^2)(E^{\mathrm{pert}} - E + \tilde{E}_{ij})\xi_{ij}^{\mathrm{WP}}$. The linear approximation for E^{pert} yields

$$\begin{aligned}\langle\langle \Delta w_{ij}^{\mathrm{WP}} \rangle\rangle &\approx \frac{\eta^2}{\sigma_{\mathrm{WP}}^4} \left\langle \left[\left(\sum_{m=1}^{M}\sum_{l=1}^{N} \frac{\partial E}{\partial w_{ml}} \xi_{ml}^{\mathrm{WP}} + \tilde{E}_{ij} \right) \xi_{ij}^{\mathrm{WP}} \right]^2 \right\rangle \\ &\quad - \eta^2 \left(\frac{\partial E}{\partial w_{ij}} \right)^2 \\ &= \eta^2 \sum_{m=1}^{M}\sum_{l=1}^{N} \left(\frac{\partial E}{\partial w_{ml}} \right)^2 + \eta^2 \left(\frac{\partial E}{\partial w_{ij}} \right)^2 \\ &\quad + \frac{\eta^2}{\sigma_{\mathrm{WP}}^2} \tilde{E}_{ij}^2.\end{aligned} \tag{A5}$$

To show that the choice of E minimizes the update noise for NP, we analogously compute the variance $\langle\langle \Delta w_{ij}^{\mathrm{NP}} \rangle\rangle$ of the NP weight updates [Eq. (6)] when adding a possibly trial- and neuron-dependent baseline term $\tilde{E}_i$, $\Delta w_{ij}^{\mathrm{NP}} = -(\eta/\sigma_{\mathrm{NP}}^2)(E^{\mathrm{pert}} - E + \tilde{E}_i) \sum_{t=1}^{T} \xi_{it}^{\mathrm{NP}} r_{jt}$:

$$\begin{aligned}\langle\langle \Delta w_{ij}^{\mathrm{NP}} \rangle\rangle &\approx \frac{\eta^2}{\sigma_{\mathrm{NP}}^4} \left\langle \left[\left(\sum_{m=1}^{M}\sum_{s=1}^{T} \frac{\partial E}{\partial y_{ms}} \xi_{ms}^{\mathrm{NP}} + \tilde{E}_i \right) \sum_{t=1}^{T} \xi_{it}^{\mathrm{NP}} r_{jt} \right]^2 \right\rangle \\ &\quad - \eta^2 \left(\frac{\partial E}{\partial w_{ij}} \right)^2 \\ &= \eta^2 \sum_{m=1}^{M}\sum_{s=1}^{T} \left(\frac{\partial E}{\partial y_{ms}} \right)^2 \sum_{t=1}^{T} r_{jt}^2 + \eta^2 \left(\frac{\partial E}{\partial w_{ij}} \right)^2 \\ &\quad + \frac{\eta^2}{\sigma_{\mathrm{NP}}^2} \tilde{E}_i^2 \sum_t r_{jt}^2.\end{aligned} \tag{A6}$$

For both algorithms, the variance is minimal if $\tilde{E} = 0$, as apparent from its quadratic occurrence with a positive prefactor.

3. Task setting

We train a linear readout $w \in \mathbb{R}^{M \times N}$ that maps N input traces $r \in \mathbb{R}^{N \times T}$ of time dimension T onto M output traces

$z \in \mathbb{R}^{M\times T}$ of the same time dimension. If not stated otherwise, input and target traces do not change between trials. The readout weights w_{ij} are trained to minimize the mean squared deviation $E = 1/(2T)\sum_{i=1}^{M}\sum_{t=1}^{T}(z_{it}-z_{it}^*)^2$ of the output z from a target z^*. The target reads, in general, $z_{it}^* = \sum_{j=1}^{N} w_{ij}^* r_{jt} + d_{it}$, where w_{ij}^* are target weights—i.e., the output error is certainly minimal for $w_{ij} = w_{ij}^*$—and d_{it} is an output component that cannot be generated by the network, since it is orthogonal to its inputs, $\sum_{t=1}^{T} d_{it} r_{jt} = 0 \;\; \forall\, i, j$. It is useful to define the (symmetric, positive semidefinite) input correlation matrix

$$S_{ij} = \frac{1}{T}\sum_{t=1}^{T} r_{it} r_{jt}. \tag{A7}$$

S can be diagonalized by a rotation $O \in \mathrm{SO}(N)$ into $D = O^T S O$ such that $D_{\mu\nu} = \alpha_\mu^2 \delta_{\mu\nu}$ is the diagonal matrix of eigenvalues $\alpha_\mu^2 \geq 0$ (where $\mu, \nu = 1, \ldots, N$). We refer to the N-dimensional (spatial) eigenvectors of S as input directions. Another useful characteristic is the autocorrelation of the inputs. We denote the autocorrelation summed over all inputs and normalized by T by

$$C_{ts} = \frac{1}{T}\sum_{j=1}^{N} r_{jt} r_{js}. \tag{A8}$$

We refer to the T-dimensional (temporal) eigenvectors of C as input components. Reading out from an input direction yields a temporal output vector parallel to the related input component. S and C have the same nonzero eigenvalues α_μ^2. (This follows, for example, from the more general fact that the products AB and BA of an $N \times T$ matrix A and a $T \times N$ matrix B have the same nonzero eigenvalues [95] by setting $A_{it} = r_{it}$ and $B_{ti} = r_{it}$.) We call the α_μ^2 input "strengths," since they equal the average strength of the μth latent input per time step or, equivalently, the average strength of all input activity read out from the μth input direction. The sum of the eigenvalues, the trace

$$\mathrm{tr}[S] \equiv \alpha_{\mathrm{tot}}^2, \tag{A9}$$

equals the total average input strength per time step α_{tot}^2. We call α_{tot}^2 the total input strength for short.

Throughout Appendix B and Supplemental Material Secs. I–VI [28], we mainly consider inputs with correlation matrices S that have N_{eff} eigenvalues equal to α^2 and all others zero (Table S1), although intermediate results can also hold for inputs with general correlations S. The correlation matrix then has the useful properties

$$S^2 = \alpha^2 S, \tag{A10}$$

$$\mathrm{tr}[S] \equiv \alpha_{\mathrm{tot}}^2 = \alpha^2 N_{\mathrm{eff}}. \tag{A11}$$

Varying the effective input dimensionality while keeping the total input strength α_{tot}^2 constant thus implies that the strengths of individual input components scale like $\alpha^2 \sim N_{\mathrm{eff}}^{-1}$.

To measure the strength of the unrealizable target component, we define a quantity α_d^2 analogous to α^2. Since $\alpha^2 = (1/N_{\mathrm{eff}})\mathrm{tr}[S] = 1/(N_{\mathrm{eff}}T)\mathrm{tr}[rr^T]$ is, in particular, the average input strength per time and latent input, we set

$$\alpha_d^2 \equiv \frac{1}{MT}\mathrm{tr}[dd^T]; \tag{A12}$$

α_d^2 is the average strength of the unrealizable component of the target per time and output.

Using the weight mismatch $W_{ij} = w_{ij} - w_{ij}^*$ and the correlation matrix S, the error function can be written as

$$\begin{aligned} E &= \frac{1}{2T}\sum_{i=1}^{M}\sum_{t=1}^{T}(z_{it}-z_{it}^*)^2 \qquad\qquad (A13)\\ &= \frac{1}{2T}\sum_{i=1}^{M}\sum_{t=1}^{T}\left(\sum_{j=1}^{N} W_{ij} r_{jt} - d_{it}\right)^2 \\ &= \frac{1}{2}\mathrm{tr}[WSW^T] + E_{\mathrm{opt}}. \qquad\qquad (A14) \end{aligned}$$

Here, E_{opt} is the lowest achievable error corresponding to zero weight mismatch:

$$E_{\mathrm{opt}} \equiv \frac{1}{2T}\mathrm{tr}[dd^T] = \frac{1}{2}M\alpha_d^2. \tag{A15}$$

We note that due to the division by T both the correlation matrix and the error no longer scale with the task duration, but the error does scale with the number M of outputs.

The choice of a quadratic error function allows one to compute the evolution of the expected error analytically. For the quadratic error function [Eqs. (8) and (A13)], the average WP and NP weight updates [Eqs. (3) and (6)] follow the gradient exactly; i.e., Eqs. (4), (A2), and (A4) hold exactly for any perturbation size:

$$\begin{aligned} \langle \Delta w_{ij}^{\mathrm{WP}} \rangle &= -\frac{\eta}{\sigma_{\mathrm{WP}}^2}\langle (E^{\mathrm{pert}} - E)\xi_{ij}^{\mathrm{WP}} \rangle = -\frac{\eta}{\sigma_{\mathrm{WP}}^2}\sum_{m=1}^{M}\sum_{k,l=1}^{N}\left(W_{mk} S_{kl} \langle \xi_{ml}^{\mathrm{WP}} \xi_{ij}^{\mathrm{WP}} \rangle + \frac{1}{2} S_{kl} \cancel{\langle \xi_{mk}^{\mathrm{WP}} \xi_{ml}^{\mathrm{WP}} \xi_{ij}^{\mathrm{WP}} \rangle} \right) \\ &= -\eta \sum_{k=1}^{N} W_{ik} S_{kj} = -\eta \frac{\partial E}{\partial w_{ij}}, \end{aligned} \tag{A16}$$

$$\begin{aligned}\langle \Delta w_{ij}^{\mathrm{NP}} \rangle &= -\frac{\eta}{\sigma_{\mathrm{NP}}^2} \sum_{t=1}^{T} \langle (E^{\mathrm{pert}} - E) \xi_{it}^{\mathrm{NP}} \rangle r_{jt} \\ &= -\frac{\eta}{\sigma_{\mathrm{WP}}^2 T} \sum_{m=1}^{M} \sum_{s,t=1}^{T} \left(\sum_{k=1}^{N} W_{mk} r_{ks} \langle \xi_{ms}^{\mathrm{NP}} \xi_{it}^{\mathrm{NP}} \rangle + \frac{1}{2} \cancel{\langle (\xi_{ms}^{\mathrm{NP}})^2 \xi_{it}^{\mathrm{NP}} \rangle} - d_{ms} \langle \xi_{ms}^{\mathrm{NP}} \xi_{it}^{\mathrm{NP}} \rangle \right) r_{jt} \\ &= -\eta \sum_{k=1}^{N} W_{ik} S_{kj} + \frac{\eta}{T} \cancel{\sum_{t=1}^{T} d_{it} r_{jt}} = -\eta \frac{\partial E}{\partial w_{ij}}. \end{aligned} \tag{A17}$$

4. Effective perturbation strength

For a fair comparison of WP and NP, we consider the output perturbations $\delta z = z^{\mathrm{pert}} - z$ that they generate. For WP, Eqs. (2) and (7) imply

$$\delta z_{it} = \sum_{j=1}^{N} \xi_{ij}^{\mathrm{WP}} r_{jt}; \tag{A18}$$

for NP, Eqs. (5) and (7) imply

$$\delta z_{it} = \xi_{it}^{\mathrm{NP}}. \tag{A19}$$

We choose σ_{WP} and σ_{NP} such that weight and node perturbations lead to the same output perturbation strength as measured by $\sigma_{\mathrm{eff}}^2 = 1/(MT)\langle \sum_{it} (\delta z_{it})^2 \rangle$, the total induced output variance per time step and output neuron:

$$\begin{aligned}\sigma_{\mathrm{eff,WP}}^2 &= \frac{1}{MT} \sum_{i=1}^{M} \sum_{t=1}^{T} \left\langle \left(\sum_{j=1}^{N} \xi_{ij}^{\mathrm{WP}} r_{jt} \right)^2 \right\rangle \\ &= \sigma_{\mathrm{WP}}^2 \cdot \alpha^2 N_{\mathrm{eff}}, \end{aligned} \tag{A20}$$

$$\sigma_{\mathrm{eff,NP}}^2 = \frac{1}{MT} \sum_{i=1}^{M} \sum_{t=1}^{T} \langle (\xi_{it}^{\mathrm{NP}})^2 \rangle = \sigma_{\mathrm{NP}}^2. \tag{A21}$$

Here, we use $\langle \xi_{ij}^{\mathrm{WP}} \xi_{mk}^{\mathrm{WP}} \rangle = \sigma_{\mathrm{WP}}^2 \delta_{im} \delta_{jk}$ and $\langle \xi_{it}^{\mathrm{NP}} \xi_{ms}^{\mathrm{NP}} \rangle = \sigma_{\mathrm{NP}}^2 \delta_{im} \delta_{ts}$. Requiring $\sigma_{\mathrm{eff,WP}}^2 \overset{!}{=} \sigma_{\mathrm{eff,NP}}^2 \equiv \sigma_{\mathrm{eff}}^2$ implies

$$\sigma_{\mathrm{NP}}^2 = \sigma_{\mathrm{eff}}^2, \qquad \sigma_{\mathrm{WP}}^2 = \frac{1}{\alpha^2 N_{\mathrm{eff}}} \cdot \sigma_{\mathrm{eff}}^2. \tag{A22}$$

We note that, although the induced output perturbations have the same variance, they follow different distributions:

$$\frac{1}{MT} \sum_{it} (\delta z_{it})^2 \sim \begin{cases} \frac{\sigma_{\mathrm{eff}}^2}{MN_{\mathrm{eff}}} \chi_{k=MN_{\mathrm{eff}}}^2 & \text{for WP,} \\ \frac{\sigma_{\mathrm{eff}}^2}{MT} \chi_{k=MT}^2 & \text{for NP,} \end{cases} \tag{A23}$$

where χ_k^2 is the chi-square distribution for k degrees of freedom.

APPENDIX B: DERIVATION OF ERROR DYNAMICS

This part starts with a derivation of the error dynamics if our tasks are learned with pure gradient descent (GD) learning. These are comparably simple and a useful benchmark. The sections thereafter provide a full derivation of the error dynamics of WP and NP learning [main text, Eqs. (9)–(14)]. Their analysis and interpretation follow in Supplemental Material Sec. I [28].

1. Error curves for gradient descent

In GD, the updates directly follow the gradient:

$$\Delta w_{ij}^{\mathrm{GD}} = -\eta \frac{\partial E}{\partial w_{ij}} = -\eta \sum_{k=1}^{N} W_{ik} S_{kj}. \tag{B1}$$

The error after such a deterministic update (which equals the expected error) is

$$\begin{aligned} E(n) &= \frac{1}{2} \mathrm{tr}\{[W(n-1) + \Delta w^{\mathrm{GD}}] \tilde{S} [W(n-1) + \Delta w^{\mathrm{GD}}]^T\} \\ &\quad + E_{\mathrm{opt}} \\ &= E(n-1) + \mathrm{tr}[W(n-1) \tilde{S} (\Delta w^{\mathrm{GD}})^T] \\ &\quad + \frac{1}{2} \mathrm{tr}[\Delta w^{\mathrm{GD}} \tilde{S} (\Delta w^{\mathrm{GD}})^T] \\ &= E(n-1) - \eta \mathrm{tr}[W \tilde{S} S W^T] + \frac{1}{2} \eta^2 \mathrm{tr}[W S \tilde{S} S W^T], \end{aligned} \tag{B2}$$

where W stands for $W(n-1)$. To facilitate the tracing of the different terms, we tag the correlation matrix that stems from the cost evaluation at trial n by a tilde, which keeps it distinguishable from those arising from the weight update Eq. (B1); the entries $\tilde{S}_{ij}$ are identical to S_{ij}. For a correlation matrix S that has N_{eff} eigenvalues equal to α^2 and all others zero (see Appendix A 3), we can use Eq. (A10) (and $\tilde{S} \equiv S$) together with $\frac{1}{2}\mathrm{tr}[WSW^T] = E(n-1) - E_{\mathrm{opt}}$ to obtain

$$\begin{aligned} E(n) &= E(n-1) - \eta \alpha^2 \mathrm{tr}[WSW^T] + \frac{1}{2} \eta^2 \alpha^4 \mathrm{tr}[WSW^T] \\ &= (1 - 2\eta\alpha^2 + \eta^2 \alpha^4) \cdot [E(n-1) - E_{\mathrm{opt}}] + E_{\mathrm{opt}}. \end{aligned} \tag{B3}$$

This leads to an exponential decay of the error to E_{opt}:

$$E(n) = [E(0) - E_{\text{opt}}]a^n + E_{\text{opt}}, \tag{B4}$$

with

$$a_{\text{GD}} = 1 - 2\eta\alpha^2 + \eta^2\alpha^4, \tag{B5}$$

which becomes zero for the optimal learning rate $\eta^* = \alpha^{-2}$ such that the task is solved in a single trial. As for WP and NP [Eqs. (11) and (B39)], learning diverges once $\eta > 2\eta^*$ where $a > 1$.

2. Error curves for weight perturbation

Since we consider only WP here, for clarity of notation we omit the specifier "WP" in ξ^{WP}, Δw^{WP}, and σ_{WP}. The error E^{pert} as a function of the perturbations ξ applied to the weights then reads

$$\begin{aligned} E^{\text{pert}} &= \frac{1}{2}\text{tr}[(W+\xi)S(W+\xi)^T] + E_{\text{opt}} \\ &= E + \text{tr}[WS\xi^T] + \frac{1}{2}\text{tr}[\xi S\xi^T], \end{aligned} \tag{B6}$$

which yields a weight update [cf. Eq. (3)]

$$\begin{aligned} \Delta w_{ij} &= -\frac{\eta}{\sigma^2}(E^{\text{pert}} - E)\xi_{ij} \\ &= -\frac{\eta}{\sigma^2}\left(\text{tr}[WS\xi^T] + \frac{1}{2}\text{tr}[\xi S\xi^T]\right)\xi_{ij}. \end{aligned} \tag{B7}$$

Averaging over ξ gives the expected error $\langle E(n)\rangle$ after the nth update:

$$\begin{aligned} \langle E(n)\rangle &= \frac{1}{2}\langle \text{tr}\{[W(n-1)+\Delta w]\tilde{S}[W(n-1)+\Delta w]^T\}\rangle + E_{\text{opt}} \\ &= \langle E(n-1)\rangle + \underbrace{\langle \text{tr}[W\tilde{S}\Delta w^T]\rangle}_{\equiv \Delta E^{\text{lin}}_{\text{update}}} \\ &\quad + \underbrace{\frac{1}{2}\langle \text{tr}[\Delta w\tilde{S}\Delta w^T]\rangle}_{\equiv \Delta E^{\text{quad}}_{\text{update}}}, \end{aligned} \tag{B8}$$

where W stands for $W(n-1)$. Here, we again tag the correlation matrix with a tilde, since it stems from the cost evaluation at trial n (cf. Appendix B 1). Using Eq. (B7) and that odd moments of ξ vanish, the first highlighted term, $\Delta E^{\text{lin}}_{\text{update}}$, which is linear in the update, gives

$$\begin{aligned} \Delta E^{\text{lin}}_{\text{update}} &= \langle \text{tr}[W\tilde{S}\Delta w^T]\rangle \\ &= -\frac{\eta}{\sigma^2}\left\langle \text{tr}\left\{ W\tilde{S}\left(\text{tr}[WS\xi^T] + \frac{1}{2}\cancel{\text{tr}[\xi S\xi^T]}\right)\xi^T\right\}\right\rangle \\ &= -\frac{\eta}{\sigma^2}\langle \text{tr}[W\tilde{S}\xi^T]\text{tr}[WS\xi^T]\rangle \\ &= -\frac{\eta}{\sigma^2}\sum_{im=1}^{M}\sum_{jklp=1}^{N} W_{ij}\tilde{S}_{jk}W_{ml}S_{lp}\cdot\langle \xi_{ik}\xi_{mp}\rangle. \end{aligned} \tag{B9}$$

Using again Eq. (B7) and that odd moments of ξ vanish, the second highlighted term, $\Delta E^{\text{quad}}_{\text{update}}$, which is quadratic in the updates, can be expanded as

$$\begin{aligned} \Delta E^{\text{quad}}_{\text{update}} &= \frac{1}{2}\langle \text{tr}[\Delta w\tilde{S}\Delta w^T]\rangle \\ &= \frac{\eta^2}{2\sigma^4}\left\langle \left(\text{tr}[WS\xi^T] + \frac{1}{2}\text{tr}[\xi S\xi^T]\right)\cdot\text{tr}[\xi\tilde{S}\xi^T]\cdot\left(\text{tr}[WS\xi^T] + \frac{1}{2}\text{tr}[\xi S\xi^T]\right)\right\rangle \\ &= \frac{\eta^2}{2\sigma^4}\langle \text{tr}[\xi\tilde{S}\xi^T]\text{tr}[WS\xi^T]^2\rangle + \frac{\eta^2}{8\sigma^4}\langle \text{tr}[\xi\tilde{S}\xi^T]\text{tr}[\xi S\xi^T]^2\rangle \\ &= \underbrace{\frac{\eta^2}{2\sigma^4}\sum_{imn=1}^{M}\sum_{jklpqr=1}^{N}\tilde{S}_{jk}W_{ml}S_{lp}W_{nq}S_{qr}\cdot\langle \xi_{ij}\xi_{ik}\xi_{mp}\xi_{nr}\rangle}_{\equiv \Delta E^{\text{quad,a}}_{\text{update}}} + \underbrace{\frac{\eta^2}{8\sigma^4}\sum_{imn=1}^{M}\sum_{jklpqr=1}^{N}\tilde{S}_{jk}S_{lp}S_{qr}\cdot\langle \xi_{ij}\xi_{ik}\xi_{ml}\xi_{mp}\xi_{nq}\xi_{nr}\rangle}_{\equiv \Delta E^{\text{quad,b}}_{\text{update}}}. \end{aligned} \tag{B10}$$

$\Delta E^{\text{quad,a}}_{\text{update}}$ depends on the weight mismatch W and originates from the gradient related part of the error signal $\Delta E_{\text{pert}} - E$, whereas $\Delta E^{\text{quad,b}}_{\text{update}}$ originates from quadratic reward noise (Supplemental Material Sec. I [28]). The moments of ξ can be computed using $\langle \xi_{ij}\xi_{mk}\rangle = \sigma^2\delta_{im}\delta_{jk}$ and Isserlis' theorem:

$$\langle \xi_{ik}\xi_{mp}\rangle = \sigma^2\delta_{im}\delta_{kp}, \tag{B11}$$

$$\langle \xi_{ij}\xi_{ik}\xi_{mp}\xi_{nr}\rangle = \sigma^4(\delta_{jk}\delta_{mn}\delta_{pr} + \delta_{im}\delta_{jp}\delta_{\text{in}}\delta_{kr} + \delta_{\text{in}}\delta_{jr}\delta_{im}\delta_{kp}), \tag{B12}$$

$$\begin{aligned}\langle \xi_{ij}\xi_{ik}\xi_{ml}\xi_{mp}\xi_{nq}\xi_{nr}\rangle = \sigma^6[&\delta_{jk}(\delta_{lp}\delta_{qr}+\delta_{mn}\delta_{lq}\delta_{pr}+\delta_{mn}\delta_{lr}\delta_{pq}) && (=\langle\xi_{ij}\xi_{ik}\rangle\langle\xi_{ml}\xi_{mp}\xi_{nq}\xi_{nr}\rangle)\\ &+\delta_{im}\delta_{jl}(\delta_{kp}\delta_{qr}+\delta_{in}\delta_{kq}\delta_{pr}+\delta_{in}\delta_{kr}\delta_{pq}) && (=\langle\xi_{ij}\xi_{ml}\rangle\langle\xi_{ik}\xi_{mp}\xi_{nq}\xi_{nr}\rangle)\\ &+\delta_{im}\delta_{jp}(\delta_{kl}\delta_{qr}+\delta_{in}\delta_{kq}\delta_{lr}+\delta_{in}\delta_{kr}\delta_{lq}) && (=\langle\xi_{ij}\xi_{mp}\rangle\langle\xi_{ik}\xi_{ml}\xi_{nq}\xi_{nr}\rangle)\\ &+\delta_{in}\delta_{jq}(\delta_{im}\delta_{kl}\delta_{pr}+\delta_{im}\delta_{kp}\delta_{lr}+\delta_{kr}\delta_{lp}) && (=\langle\xi_{ij}\xi_{nq}\rangle\langle\xi_{ik}\xi_{ml}\xi_{mp}\xi_{nr}\rangle)\\ &+\delta_{in}\delta_{jr}(\delta_{im}\delta_{kl}\delta_{pq}+\delta_{im}\delta_{kp}\delta_{lq}+\delta_{kq}\delta_{lp})]. && (=\langle\xi_{ij}\xi_{nr}\rangle\langle\xi_{ik}\xi_{ml}\xi_{mp}\xi_{nq}\rangle)\end{aligned} \tag{B13}$$

Inserting this into Eqs. (B9) and (B10) and performing the summations is partially lengthy but straightforward. It results in

$$\Delta E^{\mathrm{lin}}_{\mathrm{update}} = -\eta\sum_{i=1}^{M}\sum_{jkl=1}^{N} W_{ij}\tilde{S}_{jk}S_{lk}W_{il} = -\eta\mathrm{tr}[W\tilde{S}SW^T], \tag{B14}$$

$$\begin{aligned}\Delta E^{\mathrm{quad,a}}_{\mathrm{update}} &= \frac{\eta^2}{2}\sum_{imn=1}^{M}\sum_{jklpqr}^{N}\tilde{S}_{jk}W_{ml}S_{lp}W_{nq}S_{qr}\cdot(\delta_{jk}\delta_{mn}\delta_{pr}+\delta_{im}\delta_{jp}\delta_{in}\delta_{kr}+\delta_{in}\delta_{jr}\delta_{im}\delta_{kp})\\ &= \frac{\eta^2}{2}M\mathrm{tr}[WS^2W^T]\mathrm{tr}[\tilde{S}]+\eta^2\mathrm{tr}[WS\tilde{S}SW^T],\end{aligned} \tag{B15}$$

$$\Delta E^{\mathrm{quad,b}}_{\mathrm{update}} = \frac{\eta^2\sigma^2}{8}(M^3\mathrm{tr}[\tilde{S}]\mathrm{tr}[S]^2+2M^2\mathrm{tr}[\tilde{S}]\mathrm{tr}[S^2]+4M^2\mathrm{tr}[\tilde{S}S]\mathrm{tr}[S]+8M\mathrm{tr}[S\tilde{S}S]). \tag{B16}$$

With this, Eq. (B8) for the evolution of expected error becomes

$$\begin{aligned}\langle E(n)\rangle &\overset{\mathrm{WP}}{=} \langle E(n-1)\rangle+\Delta E^{\mathrm{lin}}_{\mathrm{update}}+\Delta E^{\mathrm{quad,a}}_{\mathrm{update}}+\Delta E^{\mathrm{quad,b}}_{\mathrm{update}}\\ &= \langle E(n-1)\rangle-\eta\mathrm{tr}[W\tilde{S}SW^T]+\frac{\eta^2}{2}M\mathrm{tr}[WS^2W^T]\mathrm{tr}[\tilde{S}]+\eta^2\mathrm{tr}[WS\tilde{S}SW^T]\\ &\quad+\frac{\eta^2\sigma^2}{8}(M^3\mathrm{tr}[\tilde{S}]\mathrm{tr}[S]^2+2M^2\mathrm{tr}[\tilde{S}]\mathrm{tr}[S^2]+4M^2\mathrm{tr}[\tilde{S}S]\mathrm{tr}[S]+8M\mathrm{tr}[S\tilde{S}S]).\end{aligned} \tag{B17}$$

The result holds for a general input correlation matrix S. In Appendix B 4, we assume same strength latent inputs (Table S1). The resulting properties of S [Eqs. (A10) and (A11)] allow one to reexpress all right-hand-side terms of Eq. (B17) in terms of $E(n-1)$ instead of W, yielding a scalar recurrence relation. We make the assumption of equally strong latent inputs also in Secs. I–III in Supplemental Material [28] (Table S1). The convergence for general input is analyzed in Supplemental Material Sec. IV [28].

3. Error curves for node perturbation

Similar to the previous section, we omit the specifier "NP" in ξ^{NP}, Δw^{NP}, and σ_{NP} in the following for notational clarity. The error E^{pert} then reads as a function of the NP ξ:

$$\begin{aligned}E^{\mathrm{pert}} &= \frac{1}{2T}\sum_{i=1}^{M}\sum_{t=1}^{T}\left(\sum_{j=1}^{N}W_{ij}r_{jt}+\xi_{it}-d_{it}\right)^2\\ &= \frac{1}{2T}\sum_{i=1}^{M}\sum_{t=1}^{T}\left[\left(\sum_{j=1}^{N}W_{ij}r_{jt}+\xi_{it}\right)^2-2\sum_{j=1}^{N}W_{ij}\cancel{r_{jt}d_{it}}-2\xi_{it}d_{it}+d_{it}^2\right]\\ &= E+\frac{1}{T}\mathrm{tr}[Wr\xi^T]+\frac{1}{2T}\mathrm{tr}[\xi\xi^T]-\frac{1}{T}\mathrm{tr}[d\xi^T].\end{aligned} \tag{B18}$$

The term $\mathrm{tr}[d\xi^T]$ here reflects the interference of learning and unrealizable targets. The other term including d, $1/(2T)\mathrm{tr}[dd^T]=E_{\mathrm{opt}}$, is only an additive constant which also occurs in GD and WP and does not enter the update equation. Equation (B18) yields a weight update [cf. Eq. (6)]:

$$\Delta w_{ij} = -\frac{\eta}{\sigma^2}(E^{\mathrm{pert}}-E)\sum_{t=1}^{T}\xi_{it}r_{jt} = -\frac{\eta}{\sigma^2 T}\left(\mathrm{tr}[Wr\xi^T]+\frac{1}{2}\mathrm{tr}[\xi\xi^T]-\mathrm{tr}[d\xi^T]\right)\sum_{t=1}^{T}\xi_{it}r_{jt}. \tag{B19}$$

Averaging over ξ gives the expected error after the update:

$$\begin{aligned}\langle E(n)\rangle &= \frac{1}{2}\langle \mathrm{tr}\{[W(n-1)+\Delta w]\tilde{S}[W(n-1)+\Delta w]^T\}\rangle + E_{\mathrm{opt}} \\ &= \langle E(n-1)\rangle + \underbrace{\langle \mathrm{tr}[W\tilde{S}\Delta w^T]\rangle}_{\equiv \Delta E^{\mathrm{lin}}_{\mathrm{update}}} + \underbrace{\frac{1}{2}\langle \mathrm{tr}[\Delta w\tilde{S}\Delta w^T]\rangle}_{\equiv \Delta E^{\mathrm{quad}}_{\mathrm{update}}},\end{aligned} \tag{B20}$$

where W stands for $W(n-1)$ and the correlation matrix that stems from the error evaluation is tagged with a tilde. Using Eq. (B19) and that odd moments of ξ vanish, $\Delta E^{\mathrm{lin}}_{\mathrm{update}}$ and $\Delta E^{\mathrm{quad}}_{\mathrm{update}}$ are

$$\begin{aligned}\Delta E^{\mathrm{lin}}_{\mathrm{update}} &= \langle \mathrm{tr}[W\tilde{S}\Delta w^T]\rangle \\ &= -\frac{\eta}{\sigma^2 T}\left\langle \mathrm{tr}\left\{ W\tilde{S}\left(\mathrm{tr}[Wr\xi^T] + \frac{1}{2}\cancel{\mathrm{tr}[\xi\xi^T]} - \mathrm{tr}[d\xi^T]\right) r\xi^T \right\}\right\rangle \\ &= -\frac{\eta}{\sigma^2 T}\langle \mathrm{tr}[W\tilde{S}r\xi^T]\mathrm{tr}[Wr\xi^T] - \mathrm{tr}[W\tilde{S}r\xi^T]\mathrm{tr}[d\xi^T]\rangle \\ &= -\frac{\eta}{\sigma^2 T}\sum_{im=1}^{M}\sum_{jkl=1}^{N}\sum_{st=1}^{T} W_{ij}\tilde{S}_{jk}W_{ml}r_{lt}r_{ks}\cdot\langle \xi_{is}\xi_{mt}\rangle + \frac{\eta}{\sigma^2 T}\sum_{im=1}^{M}\sum_{jk=1}^{N}\sum_{st=1}^{T} W_{ij}\tilde{S}_{jk}r_{ks}d_{mt}\cdot\langle \xi_{is}\xi_{mt}\rangle,\end{aligned} \tag{B21}$$

$$\begin{aligned}\Delta E^{\mathrm{quad}}_{\mathrm{update}} &= \frac{1}{2}\langle \mathrm{tr}[\Delta w\tilde{S}\Delta w^T]\rangle \\ &= \frac{\eta^2}{2\sigma^4 T^2}\left\langle \mathrm{tr}[\xi r^T\tilde{S}r\xi^T]\left(\mathrm{tr}[Wr\xi^T]^2 + \frac{1}{4}\mathrm{tr}[\xi\xi^T]^2 + \mathrm{tr}[d\xi^T]^2 - 2\mathrm{tr}[Wr\xi^T]\mathrm{tr}[d\xi^T]\right)\right\rangle \\ &= \frac{\eta^2}{2\sigma^4 T^2}\sum_{imn=1}^{M}\sum_{jklp=1}^{N}\sum_{stuv=1}^{T} r_{js}\tilde{S}_{jk}r_{kt}W_{ml}r_{lu}W_{np}r_{pv}\cdot\langle \xi_{is}\xi_{it}\xi_{mu}\xi_{nv}\rangle \quad (\Delta E^{\mathrm{quad,a}}_{\mathrm{update}})\end{aligned} \tag{B22}$$

$$+\frac{\eta^2}{8\sigma^4 T^2}\sum_{imn=1}^{M}\sum_{jk=1}^{N}\sum_{stuv=1}^{T} r_{js}\tilde{S}_{jk}r_{kt}\cdot\langle \xi_{is}\xi_{it}\xi_{mu}\xi_{mu}\xi_{nv}\xi_{nv}\rangle \qquad (\Delta E^{\mathrm{quad,b}}_{\mathrm{update}}) \tag{B23}$$

$$+\frac{\eta^2}{2\sigma^4 T^2}\sum_{imn=1}^{M}\sum_{jk=1}^{N}\sum_{stuv=1}^{T} r_{js}\tilde{S}_{jk}r_{kt}d_{mu}d_{nv}\cdot\langle \xi_{is}\xi_{it}\xi_{mu}\xi_{nv}\rangle \qquad (\Delta E^{\mathrm{quad,c}}_{\mathrm{update}}) \tag{B24}$$

$$-\frac{\eta^2}{\sigma^4 T^2}\sum_{imn=1}^{M}\sum_{jkl=1}^{N}\sum_{stuv=1}^{T} r_{js}\tilde{S}_{jk}r_{kt}W_{ml}r_{lu}d_{nv}\cdot\langle \xi_{is}\xi_{it}\xi_{mu}\xi_{nv}\rangle \qquad (\Delta E^{\mathrm{quad,d}}_{\mathrm{update}}). \tag{B25}$$

Like for WP, $\Delta E^{\mathrm{quad,a}}_{\mathrm{update}}$ depends on the weight mismatch W and originates from the gradient-related part of the error signal $\Delta E_{\mathrm{pert}} - E$, whereas $\Delta E^{\mathrm{quad,b}}_{\mathrm{update}}$ originates from quadratic reward noise (Supplemental Material Sec. I [28]). $\Delta E^{\mathrm{quad,c}}_{\mathrm{update}}$ and $\Delta E^{\mathrm{quad,d}}_{\mathrm{update}}$ stem from the reward noise due to coupling to unrealizable target components. The moments of ξ can again be computed from $\langle \xi_{is}\xi_{mt}\rangle = \sigma^2\delta_{im}\delta_{st}$ and Isserlis' theorem:

$$\langle \xi_{is}\xi_{mt}\rangle = \sigma^2\delta_{im}\delta_{st}, \tag{B26}$$

$$\langle \xi_{is}\xi_{it}\xi_{mu}\xi_{nv}\rangle = \sigma^4(\delta_{st}\delta_{mn}\delta_{uv} + \delta_{im}\delta_{su}\delta_{in}\delta_{tv} + \delta_{in}\delta_{sv}\delta_{im}\delta_{tu}), \tag{B27}$$

$$\begin{aligned}\langle \xi_{is}\xi_{it}\xi_{mu}\xi_{mu}\xi_{nv}\xi_{nv}\rangle &= \sigma^6[\delta_{st}(1+2\delta_{mn}\delta_{uv}) && (=\langle \xi_{is}\xi_{it}\rangle\langle \xi_{mu}\xi_{mu}\xi_{nv}\xi_{nv}\rangle)\\ &\quad + 2\delta_{im}\delta_{su}(\delta_{im}\delta_{tu}+2\delta_{in}\delta_{tv}\delta_{mn}\delta_{uv}) && (=2\langle \xi_{is}\xi_{mu}\rangle\langle \xi_{it}\xi_{mu}\xi_{nv}\xi_{nv}\rangle)\\ &\quad + 2\delta_{in}\delta_{sv}(\delta_{in}\delta_{tv}+2\delta_{im}\delta_{tu}\delta_{mn}\delta_{uv})] && (=2\langle \xi_{is}\xi_{nv}\rangle\langle \xi_{it}\xi_{mu}\xi_{mu}\xi_{nv}\rangle)\\ &= \sigma^6[\delta_{st}(1+2\delta_{mn}\delta_{uv})+8\delta_{imn}\delta_{stuv}+2\delta_{im}\delta_{su}\delta_{tu}+2\delta_{in}\delta_{sv}\delta_{tv}]. \end{aligned} \tag{B28}$$

Inserting this into Eqs. (B21)–(B25) gives after a partially lengthy but straightforward computation

$$\Delta E^{\mathrm{lin}}_{\mathrm{update}} = -\eta \mathrm{tr}[W\tilde{S}SW^T], \tag{B29}$$

$$\Delta E^{\mathrm{quad,a}}_{\mathrm{update}} = \frac{\eta^2}{2} M \mathrm{tr}[WSW^T]\mathrm{tr}[\tilde{S}S] + \eta^2 \mathrm{tr}[WS\tilde{S}SW^T], \tag{B30}$$

$$\Delta E^{\mathrm{quad,b}}_{\mathrm{update}} = \frac{\eta^2\sigma^2}{8T} \mathrm{tr}[\tilde{S}S]\cdot(M^3T^2+6M^2T+8M), \tag{B31}$$

$$\Delta E^{\mathrm{quad,c}}_{\mathrm{update}} = \frac{\eta^2}{2T} M \mathrm{tr}[\tilde{S}S]\mathrm{tr}[dd^T], \tag{B32}$$

$$\Delta E^{\mathrm{quad,d}}_{\mathrm{update}} = 0. \tag{B33}$$

With this, Eq. (B20) for the evolution of expected reward becomes

$$\begin{aligned}\langle E(n)\rangle &\overset{\mathrm{NP}}{=} \langle E(n-1)\rangle + \Delta E^{\mathrm{lin}}_{\mathrm{update}} + \Delta E^{\mathrm{quad,a}}_{\mathrm{update}} + \Delta E^{\mathrm{quad,b}}_{\mathrm{update}} + \Delta E^{\mathrm{quad,c}}_{\mathrm{update}}\\ &= \langle E(n-1)\rangle - \eta \mathrm{tr}[W\tilde{S}SW^T] + \frac{\eta^2}{2} M \mathrm{tr}[WSW^T]\mathrm{tr}[\tilde{S}S] + \eta^2 \mathrm{tr}[WS\tilde{S}SW^T]\\ &\quad + \frac{\eta^2\sigma^2_{\mathrm{NP}}}{8T} \mathrm{tr}[\tilde{S}S]\cdot(M^3T^2+6M^2T+8M) + \eta^2 M \mathrm{tr}[\tilde{S}S]\cdot\frac{1}{2T}\mathrm{tr}[dd^T]. \end{aligned} \tag{B34}$$

As for WP, this result holds for any correlation matrix S.

4. Error curves for equally strong input components

In this section, we consider the case that there are N_{eff} latent inputs of equal strength; see Appendix A 3. Using Eqs. (A10) and (A11) and $\tilde{S} = S$, we first simplify the evolution equation of WP's expected error [Eq. (B17)]. By identifying occurrences of $\frac{1}{2}\mathrm{tr}[WSW^T]$ and replacing them with $\langle E(n-1)\rangle - E_{\mathrm{opt}}$ [Eq. (A14)], we obtain a linear recurrence relation:

$$\begin{aligned}\langle E(n)\rangle &\overset{\mathrm{WP}}{=} \langle E(n-1)\rangle - \eta \mathrm{tr}[W\tilde{S}SW^T] + \frac{\eta^2}{2} M \mathrm{tr}[WS^2W^T]\mathrm{tr}[\tilde{S}] + \eta^2 \mathrm{tr}[WS\tilde{S}SW^T]\\ &\quad + \frac{\eta^2\sigma^2_{\mathrm{WP}}}{8}(M^3\mathrm{tr}[\tilde{S}]\mathrm{tr}[S]^2 + 2M^2\mathrm{tr}[\tilde{S}]\mathrm{tr}[S^2] + 4M^2\mathrm{tr}[\tilde{S}S]\mathrm{tr}[S] + 8M\mathrm{tr}[S\tilde{S}S])\\ &= [1-2\eta\alpha^2+\eta^2\alpha^4(MN_{\mathrm{eff}}+2)]\cdot[\langle E(n-1)\rangle - E_{\mathrm{opt}}]\\ &\quad + \frac{1}{8}\eta^2\sigma^2_{\mathrm{WP}}\alpha^6\cdot(M^3N^3_{\mathrm{eff}} + 6M^2N^2_{\mathrm{eff}} + 8MN_{\mathrm{eff}}) + E_{\mathrm{opt}}. \end{aligned} \tag{B35}$$

The evolution equation of NP's expected error [Eq. (B34)] similarly simplifies to a linear recurrence relation:

$$\begin{aligned}\langle E(n)\rangle &\overset{\mathrm{NP}}{=} \langle E(n-1)\rangle - \eta \mathrm{tr}[W\tilde{S}SW^T] + \frac{\eta^2}{2} M \mathrm{tr}[WSW^T]\mathrm{tr}[\tilde{S}S] + \eta^2 \mathrm{tr}[WS\tilde{S}SW^T]\\ &\quad + \frac{\eta^2\sigma^2_{\mathrm{NP}}}{8T} \mathrm{tr}[\tilde{S}S]\cdot(M^3T^2+6M^2T+8M) + \eta^2 M \mathrm{tr}[\tilde{S}S]\cdot\frac{1}{2T}\mathrm{tr}[dd^T]\\ &= [1-2\eta\alpha^2+\eta^2\alpha^4(MN_{\mathrm{eff}}+2)]\cdot[\langle E(n-1)\rangle - E_{\mathrm{opt}}]\\ &\quad + \frac{1}{8}\eta^2\sigma^2_{\mathrm{NP}}\alpha^4\cdot\left(M^3N_{\mathrm{eff}}T + 6M^2N_{\mathrm{eff}} + 8M\frac{N_{\mathrm{eff}}}{T}\right) + \eta^2\alpha^4 MN_{\mathrm{eff}}\cdot E_{\mathrm{opt}} + E_{\mathrm{opt}}. \end{aligned} \tag{B36}$$

Both Eqs. (B35) and (B36) have the form

$$\langle E(n)\rangle = [\langle E(n-1)\rangle - E_{\text{opt}}]\cdot a + b + E_{\text{opt}}, \tag{B37}$$

with (different) constant parameters a and b. The convergence factor a characterizes the convergence speed and b the per-update increase in error. The recurrence relation can be solved straightforwardly by iteration, by expanding b to $[b/(1-a)](1-a)$, shifting $b/(1-a)+E_{\text{opt}}$ to the equation's left-hand side, and considering $\langle E(n)\rangle - E_{\text{opt}} - b/(1-a)$ as a recurrently specified variable:

$$\langle E(n)\rangle = \left(E(0) - \frac{b}{1-a} - E_{\text{opt}}\right)\cdot a^n + \frac{b}{1-a} + E_{\text{opt}}. \tag{B38}$$

Another possibility is to consider $\langle E(n)\rangle - E_{\text{opt}}$ as a recurrently specified variable and to observe that the iteration gives rise to a finite geometric series that yields $(1-a^n)/(1-a)b$. The values of the constants are

$$a = 1 - 2\eta\alpha^2 + \eta^2\alpha^4(MN_{\text{eff}} + 2), \tag{B39}$$

$$b_{\text{WP}} = \frac{1}{8}\eta^2\sigma_{\text{WP}}^2\alpha^6\cdot(M^3N_{\text{eff}}^3 + 6M^2N_{\text{eff}}^2 + 8MN_{\text{eff}}), \tag{B40}$$

$$b_{\text{NP}} = \frac{1}{8}\eta^2\sigma_{\text{NP}}^2\alpha^4\cdot\left(M^3N_{\text{eff}}T + 6M^2N_{\text{eff}} + 8M\frac{N_{\text{eff}}}{T}\right) + \eta^2\alpha^4MN_{\text{eff}}\cdot E_{\text{opt}}. \tag{B41}$$

Finite b due to finite perturbation size σ^2 causes a finite residual error $b/(1-a)$ in Eq. (B38) even if E_{opt} is zero. To enable a fair comparison, σ_{WP} and σ_{NP} are chosen such that they lead to output perturbations δz of the same strength; see Appendix A4. Expressing σ_{WP}^2 and σ_{NP}^2 through the strength σ_{eff}^2 of the output perturbation that they generate [Eq. (A22)], the constants b become

$$b_{\text{WP}} = \frac{1}{8}\eta^2\sigma_{\text{eff}}^2\alpha^4\cdot(M^3N_{\text{eff}}^2 + 6M^2N_{\text{eff}} + 8M), \tag{B42}$$

$$b_{\text{NP}} = \underbrace{\frac{1}{8}\eta^2\sigma_{\text{eff}}^2\alpha^4\cdot\left(M^3N_{\text{eff}}T + 6M^2N_{\text{eff}} + 8M\frac{N_{\text{eff}}}{T}\right)}_{} + \underbrace{\eta^2\alpha^4MN_{\text{eff}}\cdot E_{\text{opt}}}_{} \tag{B43}$$

$$\equiv \qquad b_{\text{NP}}^{\text{quad}} \qquad + \qquad b_{\text{NP}}^{\text{lin}}, \tag{B44}$$

where the splitting of b_{NP} into $b_{\text{NP}}^{\text{quad}}$ and $b_{\text{NP}}^{\text{lin}}$ is motivated in the next part. a is independent of the perturbation size and the same for WP and NP.

5. Optimal learning rate

The optimal learning rate and convergence factor are obtained by minimizing the quadratic function a as a function of η with respect to η:

$$\eta^* = \underset{\eta}{\text{argmin}}\ a(\eta) = \frac{1}{(MN_{\text{eff}}+2)\alpha^2}, \tag{B45}$$

$$a^* = \min_{\eta} a(\eta) = 1 - \frac{1}{MN_{\text{eff}}+2} = a(\eta^*). \tag{B46}$$

Learning diverges for $\eta \to 2\eta^*$ because then $a \to 1$:

$$a(2\eta^*) = 1 - \frac{4}{MN_{\text{eff}}+2} + 4\frac{MN_{\text{eff}}+2}{(MN_{\text{eff}}+2)^2} = 1. \tag{B47}$$

[1] D. M. Wolpert, J. Diedrichsen, and J. R. Flanagan, *Principles of Sensorimotor Learning*, Nat. Rev. Neurosci. **12**, 739 (2011).

[2] J. W. Krakauer and P. Mazzoni, *Human Sensorimotor Learning: Adaptation, Skill, and beyond*, Curr. Opin. Neurobiol. **21**, 636 (2011).

[3] T. P. Lillicrap, A. Santoro, L. Marris, C. J. Akerman, and G. Hinton, *Backpropagation and the Brain*, Nat. Rev. Neurosci. **21**, 335 (2020).

[4] C. Klos, Y. F. K. Kossio, S. Goedeke, A. Gilra, and R.-M. Memmesheimer, *Dynamical Learning of Dynamics*, Phys. Rev. Lett. **125**, 088103 (2020).

[5] P. Dayan and L. Abbott, *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems* (MIT, Cambridge, MA, 2001).

[6] R. S. Sutton, *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning Series)* (Bradford, Cambridge, MA, 2018).

[7] A. Dembo and T. Kailath, *Model-Free Distributed Learning*, IEEE Trans. Neural Networks **1**, 58 (1990).

[8] G. Cauwenberghs, *A Fast Stochastic Error-Descent Algorithm for Supervised Learning and Optimization*, in *Advances in Neural Information Processing Systems* (Morgan Kaufmann, Burlington, 1993), Vol. 5, pp. 244–251.

[9] J. Werfel, X. Xie, and H. S. Seung, *Learning Curves for Stochastic Gradient Descent in Linear Feedforward Networks*, Neural Comput. **17**, 2699 (2005).

[10] I. R. Fiete and H. S. Seung, *Gradient Learning in Spiking Neural Networks by Dynamic Perturbation of Conductances*, Phys. Rev. Lett. **97**, 048104 (2006).

[11] R. J. Williams, *Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning*, Mach. Learn. **8**, 229 (1992).
[12] R. Legenstein, S. M. Chase, A. B. Schwartz, and W. Maass, *A Reward-Modulated Hebbian Learning Rule Can Explain Experimentally Observed Network Reorganization in a Brain Control Task*, J. Neurosci. **30**, 8400 (2010).
[13] G. M. Hoerzer, R. Legenstein, and W. Maass, *Emergence of Complex Computational Structures from Chaotic Neural Networks through Reward-Modulated Hebbian Learning*, Cereb. Cortex **24**, 677 (2014).
[14] T. Miconi, *Biologically Plausible Learning in Recurrent Neural Networks Reproduces Neural Dynamics Observed during Cognitive Tasks*, eLife **6**, e20899 (2017).
[15] M. Jabri and B. Flower, *Weight Perturbation: An Optimal Architecture and Learning Technique for Analog VLSI Feedforward and Recurrent Multilayer Networks*, IEEE Trans. Neural Networks **3**, 154 (1992).
[16] H. Saito, K. Katahira, K. Okanoya, and M. Okada, *Statistical Mechanics of Structural and Temporal Credit Assignment Effects on Learning in Neural Networks*, Phys. Rev. E **83**, 051125 (2011).
[17] I. R. Fiete, M. S. Fee, and H. S. Seung, *Model of Birdsong Learning Based on Gradient Estimation by Dynamic Perturbation of Neural Conductances*, J. Neurophysiol. **98**, 2038 (2007).
[18] R. Mooney, J. Prather, and T. Roberts, *Neurophysiology of Birdsong Learning*, in *Learning and Memory: A Comprehensive Reference* (Elsevier, New York, 2008), pp. 441–474.
[19] P. Gao, E. Trautmann, B. Yu, G. Santhanam, S. Ryu, K. Shenoy, and S. Ganguli, *A Theory of Multineuronal Dimensionality, Dynamics and Measurement*, bioRxiv.
[20] M. C. Tresch and A. Jarc, *The Case for and against Muscle Synergies*, Curr. Opin. Neurobiol. **19**, 601 (2009).
[21] J. A. Gallego, M. G. Perich, L. E. Miller, and S. A. Solla, *Neural Manifolds for the Control of Movement*, Neuron **94**, 978 (2017).
[22] K. Doya and T. J. Sejnowski, *A Computational Model of Birdsong Learning by Auditory Experience and Auditory Feedback*, in *Central Auditory Processing and Neural Modeling* (Springer, New York, 1998), pp. 77–88.
[23] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal Dynamics—From Single Neurons to Networks and Models of Cognition* (Cambridge University Press, Cambridge, England, 2014).
[24] B. Widrow and M. A. Lehr, *30 Years of Adaptive Neural Networks: Perceptron, Madaline, and Backpropagation*, Proc. IEEE **78**, 1415 (1990).
[25] M. London and M. Häusser, *Dendritic Computation*, Annu. Rev. Neurosci. **28**, 503 (2005).
[26] R.-M. Memmesheimer, *Quantitative Prediction of Intermittent High-Frequency Oscillations in Neural Networks with Supralinear Dendritic Interactions*, Proc. Natl. Acad. Sci. U.S.A. **107**, 11092 (2010).
[27] D. Breuer, M. Timme, and R.-M. Memmesheimer, *Statistical Physics of Neural Systems with Non-additive Dendritic Coupling*, Phys. Rev. X **4**, 011053 (2014).
[28] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevX.13.021006 for detailed derivations, further analysis, and accompanying simulations.
[29] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT, Cambridge, MA, 2016), http://www.deeplearningbook.org.
[30] F. Zenke, G. Hennequin, and W. Gerstner, *Synaptic Plasticity in Neural Networks Needs Homeostasis with a Fast Rate Detector*, PLoS Comput. Biol. **9**, e1003330 (2013).
[31] N. E. Ziv and N. Brenner, *Synaptic Tenacity or Lack Thereof: Spontaneous Remodeling of Synapses*, Trends Neurosci. **41**, 89 (2018).
[32] R. Kawai, T. Markman, R. Poddar, R. Ko, A. L. Fantana, A. K. Dhawale, A. R. Kampff, and B. P. Ölveczky, *Motor Cortex Is Required for Learning but not for Executing a Motor Skill*, Neuron **86**, 800 (2015).
[33] K. Katahira, T. Cho, K. Okanoya, and M. Okada, *Optimal Node Perturbation in Linear Perceptrons with Uncertain Eligibility Trace*, Neural Netw. **23**, 219 (2010).
[34] M. Beyeler, E. L. Rounds, K. D. Carlson, N. Dutt, and J. L. Krichmar, *Neural Correlates of Sparse Coding and Dimensionality Reduction*, PLoS Comput. Biol. **15**, e1006908 (2019).
[35] R. Hahnloser, A. Kozhevnikov, and M. Fee, *An Ultra-sparse Code Underlies the Generation of Neural Sequences in a Songbird*, Nature (London) **419**, 65 (2002).
[36] R. Q. Quiroga, L. Reddy, G. Kreiman, C. Koch, and I. Fried, *Invariant Visual Representation by Single Neurons in the Human Brain*, Nature (London) **435**, 1102 (2005).
[37] D. V. Buonomano and M. M. Merzenich, *Temporal Information Transformed into a Spatial Code by a Neural Network with Realistic Properties*, Science **267**, 1028 (1995).
[38] P. F. Dominey, *Complex Sensory-Motor Sequence Learning Based on Recurrent State Representation and Reinforcement Learning*, Biol. Cybern. **73**, 265 (1995).
[39] W. Maass, T. Natschläger, and H. Markram, *A Model for Real Time Computation in Generic Microcircuits*, in *Advances in Neural Information Processing Systems*, edited by S. Becker, S. Thrün, and K. Obermayer (MIT, Cambridge, MA, 2003).
[40] D. Sussillo and L. F. Abbott, *Generating Coherent Patterns of Activity from Chaotic Neural Networks*, Neuron **63**, 544 (2009).
[41] D. Thalmeier, M. Uhlmann, H. J. Kappen, and R.-M. Memmesheimer, *Learning Universal Computations with Spikes*, PLoS Comput. Biol. **12**, e1004895 (2016).
[42] R. Pyle and R. Rosenbaum, *A Reservoir Computing Model of Reward-Modulated Motor Learning and Automaticity*, Neural Comput. **31**, 1430 (2019).
[43] G. Hennequin, T. P. Vogels, and W. Gerstner, *Optimal Control of Transient Dynamics in Balanced Networks Supports Generation of Complex Movements*, Neuron **82**, 1394 (2014).
[44] M. F. Bear, B. W. Connors, and M. A. Paradiso, *Neuroscience—Exploring the Brain* (Wolters Kluwer, Philadelphia, 2016).
[45] N. Y. Masse, G. R. Yang, H. F. Song, X.-J. Wang, and D. J. Freedman, *Circuit Mechanisms for the Maintenance*

and Manipulation of Information in Working Memory, Nat. Neurosci. **22**, 1159 (2019).
[46] A. K. Dhawale, M. A. Smith, and B. P. Ölveczky, *The Role of Variability in Motor Learning*, Annu. Rev. Neurosci. **40**, 479 (2017).
[47] J. P. Cunningham and B. M. Yu, *Dimensionality Reduction for Large-Scale Neural Recordings*, Nat. Neurosci. **17**, 1500 (2014).
[48] L. F. Abbott, K. Rajan, and H. Sompolinsky, *The Dynamic Brain: An Exploration of Neuronal Variability and Its Functional Significance* (Oxford University, New York, 2011), Chap. Interactions between Intrinsic and Stimulus-Evoked Activity in Recurrent Neural Networks, pp. 65–82.
[49] C. J. Cueva, A. Saez, E. Marcos, A. Genovesio, M. Jazayeri, R. Romo, C. D. Salzman, M. N. Shadlen, and S. Fusi, *Low-Dimensional Dynamics for Working Memory and Time Encoding*, Proc. Natl. Acad. Sci. U.S.A. **117**, 23021 (2020).
[50] M. Russo, M. D'Andola, A. Portone, F. Lacquaniti, and A. d'Avella, *Dimensionality of Joint Torques and Muscle Patterns for Reaching*, Front. Comput. Neurosci. **8**, 00024 (2014).
[51] A. Destexhe, M. Rudolph, and D. Paré, *The High-Conductance State of Neocortical Neurons In Vivo*, Nat. Rev. Neurosci. **4**, 739 (2003).
[52] J. D. Murray, A. Bernacchia, D. J. Freedman, R. Romo, J. D. Wallis, X. Cai, C. Padoa-Schioppa, T. Pasternak, H. Seo, D. Lee, and X.-J. Wang, *A Hierarchy of Intrinsic Timescales across Primate Cortex*, Nat. Neurosci. **17**, 1661 (2014).
[53] T. Teşileanu, B. Ölveczky, and V. Balasubramanian, *Rules and Mechanisms for Efficient Two-Stage Learning in Neural Circuits*, eLife **6**, 20944 (2017).
[54] J. M. Murray and G. S. Escola, *Learning Multiple Variable-Speed Sequences in Striatum via Cortical Tutoring*, eLife **6**, 26084 (2017).
[55] A. K. Dhawale, R. Poddar, S. B. Wolff, V. A. Normand, E. Kopelowitz, and B. P. Ölveczky, *Automated Long-Term Recording and Analysis of Neural Activity in Behaving Animals*, eLife **6**, 27702 (2017).
[56] S. Cheng, *The CRISP Theory of Hippocampal Function in Episodic Memory*, Front. Neural Circuits **7**, 88 (2013).
[57] A. Maes, M. Barahona, and C. Clopath, *Learning Spatio-temporal Signals Using a Recurrent Spiking Network that Discretizes Time*, PLoS Comput. Biol. **16**, e1007606 (2020).
[58] H. S. Seung, *Learning in Spiking Neural Networks by Reinforcement of Stochastic Synaptic Transmission*, Neuron **40**, 1063 (2003).
[59] Ł. Kuśmierz, T. Isomura, and T. Toyoizumi, *Learning with Three Factors: Modulating Hebbian Plasticity with Errors*, Curr. Opin. Neurobiol. **46**, 170 (2017).
[60] R. L. Redondo and R. G. M. Morris, *Making Memories Last: The Synaptic Tagging and Capture Hypothesis*, Nat. Rev. Neurosci. **12**, 17 (2011).
[61] J. N. Reynolds and J. R. Wickens, *Dopamine-Dependent Plasticity of Corticostriatal Synapses*, Neural Netw. **15**, 507 (2002).
[62] Q. Zhou and M. Poo, *Reversal and Consolidation of Activity-Induced Synaptic Modifications*, Trends Neurosci. **27**, 378 (2004).
[63] J. Luboeinski and C. Tetzlaff, *Memory Consolidation and Improvement by Synaptic Tagging and Capture in Recurrent Neural Networks*, Commun. Biol. **4**, 275 (2021).
[64] E. M. Izhikevich, *Solving the Distal Reward Problem through Linkage of STDP and Dopamine Signaling*, Cereb. Cortex **17**, 2443 (2007).
[65] M. A. Farries and A. L. Fairhall, *Reinforcement Learning with Modulated Spike Timing–Dependent Synaptic Plasticity*, J. Neurophysiol. **98**, 3648 (2007).
[66] X. Yao, *Evolving Artificial Neural Networks*, Proc. IEEE **87**, 1423 (1999).
[67] D. Floreano, P. Dürr, and C. Mattiussi, *Neuroevolution: From Architectures to Learning*, Evolut. Intell. **1**, 47 (2008).
[68] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever, *Evolution Strategies as a Scalable Alternative to Reinforcement Learning*, arXiv:1703.03864.
[69] J. T. Trachtenberg, B. E. Chen, G. W. Knott, G. Feng, J. R. Sanes, E. Welker, and K. Svoboda, *Long-Term In Vivo Imaging of Experience-Dependent Synaptic Plasticity in Adult Cortex*, Nature (London) **420**, 788 (2002).
[70] A. J. Holtmaat, J. T. Trachtenberg, L. Wilbrecht, G. M. Shepherd, X. Zhang, G. W. Knott, and K. Svoboda, *Transient and Persistent Dendritic Spines in the Neocortex In Vivo*, Neuron **45**, 279 (2005).
[71] N. Yasumatsu, M. Matsuzaki, T. Miyazaki, J. Noguchi, and H. Kasai, *Principles of Long-Term Dynamics of Dendritic Spines*, J. Neurosci. **28**, 13592 (2008).
[72] A. Minerbi, R. Kahana, L. Goldfeld, M. Kaufman, S. Marom, and N. E. Ziv, *Long-Term Relationships between Synaptic Tenacity, Synaptic Remodeling, and Network Activity*, PLoS Biol. **7**, e1000136 (2009).
[73] A. Knoblauch, *The Role of Structural Plasticity and Synaptic Consolidation for Memory and Amnesia in a Model of Cortico-Hippocampal Interplay*, in *Connectionist Models of Behaviour and Cognition II* (World Scientific, Singapore, 2009).
[74] M. Deger, M. Helias, S. Rotter, and M. Diesmann, *Spike-Timing Dependence of Structural Plasticity Explains Cooperative Synapse Formation in the Neocortex*, PLoS Comput. Biol. **8**, e1002689 (2012).
[75] A. Knoblauch, E. Körner, U. Körner, and F. T. Sommer, *Structural Synaptic Plasticity Has High Memory Capacity and Can Explain Graded Amnesia, Catastrophic Forgetting, and the Spacing Effect*, PLoS One **9**, e96485 (2014).
[76] M. Fauth, F. Wörgötter, and C. Tetzlaff, *The Formation of Multi-synaptic Connections by the Interaction of Synaptic and Structural Plasticity and Their Functional Consequences*, PLoS Comput. Biol. **11**, e1004031 (2015).
[77] A. Knoblauch, *Impact of Structural Plasticity on Memory Formation and Decline*, in *The Rewiring Brain* (Elsevier, New York, 2017), pp. 361–386.
[78] M. J. Fauth and M. C. van Rossum, *Self-Organized Reactivation Maintains and Reinforces Memories despite Synaptic Turnover*, eLife **8**, 43717 (2019).
[79] D. Kappel, R. Legenstein, S. Habenschuss, M. Hsieh, and W. Maass, *A Dynamic Connectome Supports the Emergence of Stable Computational Function of Neural Circuits through Reward-Based Learning*, eNeuro **5**, 0301 (2018).
[80] P. Suszynski and P. Wawrzynski, *Learning Population of Spiking Neural Networks with Perturbation of*

Conductances, in *Proceedings of the 2013 International Joint Conference on Neural Networks (IJCNN)* (IEEE, New York, 2013).

[81] X. Xie and H. S. Seung, *Learning in Neural Networks by Reinforcement of Irregular Spiking*, Phys. Rev. E **69**, 041909 (2004).

[82] T. Cho, K. Katahira, K. Okanoya, and M. Okada, *Node Perturbation Learning without Noiseless Baseline*, Neural Netw. **24**, 267 (2011).

[83] P. Mazzoni, R. A. Andersen, and M. I. Jordan, *A More Biologically Plausible Learning Rule for Neural Networks*, Proc. Natl. Acad. Sci. U.S.A. **88**, 4433 (1991).

[84] R. Darshan, A. Leblois, and D. Hansel, *Interference and Shaping in Sensorimotor Adaptations with Rewards*, PLoS Comput. Biol. **10**, e1003377 (2014).

[85] E. Vasilaki, S. Fusi, X.-J. Wang, and W. Senn, *Learning Flexible Sensori-motor Mappings in a Complex Network*, Biol. Cybern. **100**, 147 (2009).

[86] K. Takiyama and M. Okada, *Maximization of Learning Speed in the Motor Cortex due to Neuronal Redundancy*, PLoS Comput. Biol. **8**, e1002348 (2012).

[87] M. N. Abdelghani, T. P. Lillicrap, and D. B. Tweed, *Sensitivity Derivatives for Flexible Sensorimotor Learning*, Neural Comput. **20**, 2085 (2008).

[88] B. B. Vladimirskiy, E. Vasilaki, R. Urbanczik, and W. Senn, *Stimulus Sampling as an Exploration Mechanism for Fast Reinforcement Learning*, Biol. Cybern. **100**, 319 (2009).

[89] J. Friedrich, R. Urbanczik, and W. Senn, *Code-Specific Learning Rules Improve Action Selection by Populations of Spiking Neurons*, Int. J. Neural Syst. **24**, 1450002 (2014).

[90] A. Payeur, J. Guerguiev, F. Zenke, B. A. Richards, and R. Naud, *Burst-Dependent Synaptic Plasticity Can Coordinate Learning in Hierarchical Circuits*, Nat. Neurosci. **24**, 1010 (2021).

[91] T. P. Lillicrap, D. Cownden, D. B. Tweed, and C. J. Akerman, *Random Synaptic Feedback Weights Support Error Backpropagation for Deep Learning*, Nat. Commun. **7**, 13276 (2016).

[92] K. Doya, *Bifurcations in the Learning of Recurrent Neural Networks*, in *Proceedings of the 1992 IEEE International Symposium on Circuits and Systems, San Diego, CA* (IEEE, New York, 1992), Vol. 6, pp. 2777–2780.

[93] M. D. Giudice, *Effective Dimensionality: A Tutorial*, Multivar. Behav. Res. **56**, 527 (2021).

[94] P. Züge and C. Klos, *Source code for simulations supplementing the article "Weight versus Node Perturbation Learning in Temporally Extended Tasks: Weight Perturbation Often Performs Similarly or Better"*, GitHub, https://github.com/PaulZOnGit/WP-versus-NP (2023).

[95] R. A. Horn and C. R. Johnson, *Matrix Analysis*, 2nd ed. (Cambridge University Press, Cambridge, England, 2012).

APPENDIX B

Supplement 1: Weight versus Node Perturbation Learning in Temporally Extended Tasks: Weight Perturbation Often Performs Similarly or Better

This appendix, 'supplement 1' [120], contains a full copy of the supplement to publication 1:

[120] **P. Züge**, C. Klos and R.-M. Memmesheimer
Supplementary material to
'Weight versus Node Perturbation Learning in Temporally Extended Tasks: Weight Perturbation Often Performs Similarly or Better'
Available at Physical Review X **13** (2023) 021006

For the contribution statement, see Chapter 3.

Supplementary Material to ‘Weight vs. Node Perturbation Learning in Temporally Extended Tasks: Weight Perturbation often Performs Similarly or Better’

Paul Züge[1], **Christian Klos**[1], **Raoul-Martin Memmesheimer**[1*]

[1]Neural Network Dynamics and Computation, Institute of Genetics, University of Bonn;
[*]rm.memmesheimer@uni-bonn.de (corresponding author)

The supplement follows the structure of the main text; main results and equations referenced in the main text are highlighted for clarity with a yellow background. The Supplementary Material (SM) has six parts, SM1-SM6, each with several sections. Further there are twelve supplementary figures, Figs. S1–S12 and two supplementary tables Tabs. 1,2. The first table, Tab. 1 below, gives an overview of the assumptions used in the different parts and sections of the supplement (and the appendix).

Part / Section	Linear networks	Repeated inputs	Same strength inputs
Appendix A: “Learning models and task principles”			
“Mean updates...”, “Dependence of...”		×	
“Task setting”, “Effective pert...”	×	×	×
Appendix B: “Derivation of error dynamics”			
“Error curves for gradient descent” – “...for node perturbation”	×	×	
“...for equally strong input components”, “Optimal learning rate”	×	×	×
Part 1: “Analysis of error dynamics”	×	×	×
Part 2: “Analysis of Weight Diffusion”	×	×	×
Part 3: “Multiple subtasks”	×		×
Part 4: “Arbitrary input strength distributions”	×	×	
Part 5: “Input and perturbation correlations”	×	×	(×)
Part 6: “Improved learning rules”			
“WP0: Assign zero credit to zero inputs”			
“Hybrid perturbation (HP): Using WP...”	×	×	(×)

Table 1. Assumptions used in the different parts of the appendix and supplement. Crosses, ×, mean that an assumption is used. If different sections of a part use different assumptions, all sections are specified. Parentheses mean that an assumption is used only in part of a section.

SM 1

Analysis of error dynamics

In order to interpret and explain the main results of main text Sec. “Error dynamics”, it is helpful to make four distinctions: 1) between task relevant and irrelevant weights, 2) between realizable and unrealizable outputs, 3) between informative linear, uninformative linear and uninformative quadratic contributions to the error signal, and 4) between update fluctuations due to a credit assignment problem and those due to reward noise. In the following sections we will work these distinctions out, considering first the error signal and then the thereby informed updates. Finally we use the concepts to explain the components and behavior of the recurrence relation that describes the resulting error evolution. As before, we consider training linear readouts to reduce a quadratic error. Inputs and targets exactly repeat each trial. We focus on networks where the first N_{eff} eigenvalues of the input correlation matrix S are equal to α^2 and all others are zero (App. B, Sec. “Error curves for equally strong input components”, main text Sec. “Theoretical analysis”).

Task relevant and irrelevant weights

For low-dimensional input, weight changes along many directions in synaptic weight space influence neither output nor error. Here we define *relevant* and *irrelevant weight space directions* and, after an input rotation that we show leaves WP and NP learning invariant, *relevant* and *irrelevant weights*. Distinguishing task-relevant and -irrelevant weights will allow us to explain why irrelevant weights diffuse for WP but not NP, and why WP nevertheless attains the same convergence speed.

Because any $S = \frac{1}{T} r r^T$ is symmetric, it can be diagonalized by a $N \times N$ rotation matrix O, $D = O^T S O$. We can therefore define a set of rotated inputs $\tilde{r} = O^T r$ ($\tilde{r}_{\mu t} = \sum_{l=1}^{N} (O^T)_{\mu l} r_{lt}$) that are mutually uncorrelated as their correlation matrix is diagonal: $\frac{1}{T} \tilde{r} \tilde{r}^T = \frac{1}{T} O^T r r^T O = O^T S O = D$. The rotation does not affect the output of our networks if the reverse rotation is applied to the readout weights $\tilde{w} = wO$,

$$z_{it} = \sum_{j=1}^{N} w_{ij} r_{jt} = \sum_{j\mu l=1}^{N} w_{ij} \underbrace{O_{j\mu} \cdot (O^T)_{\mu l}}_{\Rightarrow \delta_{jl}} r_{lt} = \sum_{\mu=1}^{N} \tilde{w}_{i\mu} \tilde{r}_{\mu t} = \tilde{z}_{it}. \tag{S1}$$

The rotational invariance holds as long as inputs sum linearly, unaffected by the (possibly nonlinear) activation functions g. WP and NP work the same before and after the rotation of inputs because the noise is Gaussian iid and thus in particular isotropic. This is also reflected by the fact that all results only depend on the traces of powers of S, see Eqs. (B17, B34), which are invariant to rotations. Up to fluctuations due to different noise realizations, a WP or NP learning network thus behaves the same as its counterpart with rotated inputs, if the initial weights of the latter are transformed with the inverse rotation.

In networks where the first N_{eff} eigenvalues of the input correlation matrix S are equal to α^2 and all others are zero, due to the equivalence of the learning dynamics for rotated and unrotated inputs, we can assume without loss of generality that the first N_{eff} inputs are orthogonal and have strength α^2, while the last $N - N_{\text{eff}}$ inputs are zero. Main text, Fig. 2a illustrates inputs with the assumed correlation matrix before and after the rotation. Because the last $N - N_{\text{eff}}$ inputs are always zero, all the $M(N - N_{\text{eff}})$ weights that connect them to the outputs are irrelevant for the task. Conversely, the $M N_{\text{eff}}$ weights that originate from the first N_{eff} inputs are task relevant; changes of these weights affect performance. More generally, irrelevant weights occur in a network with rotated inputs whenever the obtained diagonal input correlation matrix D contains zero diagonal entries. These weights correspond to irrelevant weight directions in the network with unrotated inputs.

Considering relevant and irrelevant weights instead of weight space directions, which are linear combinations of individual weights, will simplify our discussion. In all of the following we will therefore without loss of generality assume rotated inputs.

Realizable and unrealizable outputs

General targets and the output perturbations of NP can contain components that are not present in the inputs and thus not realizable. Here we define these components and describe their different effects in WP and NP learning. This will be crucial to understand one part of the difference between the final errors that WP and NP can obtain.

If there is a single linear readout, the inputs (i.e. the temporal input vectors) span the space of outputs (of temporal output vectors) that can be realized by adjusting the weights. If the inputs are effectively N_{eff}-dimensional as above, that space is also N_{eff}-dimensional. For M outputs it is $M N_{\text{eff}}$-dimensional. The full output space can thus be split into an $M N_{\text{eff}}$-dimensional subspace of *realizable outputs*, and an orthogonal $M(T - N_{\text{eff}})$-dimensional subspace of *unrealizable outputs*.

Because WP perturbs the weights, the resulting perturbations of the outputs are always reproducible through a weight update. NP, on the other hand, perturbs all MT dimensions of the full output space equally, such that only a fraction of N_{eff}/T of its perturbation's variance falls into the realizable subspace. However, only the realizable components of an output perturbation can be used to inform weight updates. If, for example, an unrealizable component improves performance, there is no way to capitalize on this by reproducing it through a weight update. As a consequence, the unrealizable output perturbation components of NP are a source of *reward noise*, which partially obscures the informative error changes due to better or worse generation of the realizable components. In particular, this lowers NP's performance if the target has an unrealizable component.

The worse performance of NP can be understood in more detail as follows: In presence of unrealizable target components, also the output error gradient will have a component in the unrealizable subspace. Unrealizable node perturbation components can "couple" to this component, i.e. they can have a nonzero projection onto it, contributing to the error E^{pert} already in linear order. To translate output perturbations into appropriate weight updates, the eligibility trace Eq. (6) projects them onto the inputs, which deletes all unrealizable perturbation components. Their contribution to $E^{\text{pert}} - E$, however, still enters the update. From the perspective of the weight updates the unrealizable output perturbation components therefore just contribute noise of unknown origin to the error change $E^{\text{pert}} - E$. This noise adds to the informative error change that results from realizable perturbations, which are reflected in the eligibility trace and translated into weight changes. These different contributions of the perturbations to the error signal and their effects on updates and error evolution will be quantitatively analyzed in the following sections.

Linear informative, linear uninformative and quadratic uninformative contributions to the error signal

In this section we distinguish the different contributions to the error signal and discuss their magnitude and scaling. This will allow us to understand the origin of the different contributions to the updates as well as their effects on the error evolution and to compare their impact for WP and NP.

The error change $\Delta E_{\text{pert}} = E^{\text{pert}} - E$ due to a perturbation can be split into a linear and a quadratic part,

$$\Delta E_{\text{pert}} = \Delta E^{\text{lin}}_{\text{pert}} + \Delta E^{\text{quad}}_{\text{pert}} . \tag{S2}$$

Higher orders do not occur due to the use of a quadratic error function (Eq. (8)).

We first consider WP, where

$$\Delta E^{\text{WP}}_{\text{pert}} = \overbrace{\text{tr}[W S (\xi^{\text{WP}})^T]}^{\equiv \Delta E^{\text{lin,WP}}_{\text{pert}}} + \overbrace{\tfrac{1}{2}\,\text{tr}[\xi^{\text{WP}} S (\xi^{\text{WP}})^T]}^{\equiv \Delta E^{\text{quad,WP}}_{\text{pert}}} \tag{S3}$$

(Eq. (B6)). The linear part of an error change due to weight perturbations can generally also be written as

$$\Delta E^{\text{lin,WP}}_{\text{pert}} = \sum_{mk} \frac{\partial E}{\partial w_{mk}} \cdot \xi^{\text{WP}}_{mk} , \tag{S4}$$

cf. Eq. (A1). This shows that $\Delta E^{\text{lin,WP}}_{\text{pert}}$ contains all the information about the error gradient employed in weight perturbation learning, namely the size of the tried perturbation's projection onto it: The update equations Eqs. (4,A2,A16) use that if we average all normalized perturbation vectors weighted by their projections onto the gradient, the result is the gradient. $\Delta E^{\text{lin}}_{\text{pert}}$ provides the employed projections. In contrast, the quadratic part $\Delta E^{\text{quad}}_{\text{pert}}$ of ΔE_{pert} does not contain such information: $\Delta E^{\text{quad}}_{\text{pert}}$ is uncorrelated with the size of the projection of ξ^{WP} onto the gradient, $\langle \Delta E^{\text{lin}}_{\text{pert}} \Delta E^{\text{quad}}_{\text{pert}} \rangle = 0$. $\Delta E^{\text{quad}}_{\text{pert}}$ therefore only adds reward noise to the learning process. When averaging over perturbations, this does not bias the resulting update, because $\Delta E^{\text{quad}}_{\text{pert}}$ is also uncorrelated with ξ^{WP}, $\langle \Delta E^{\text{quad}}_{\text{pert}} \xi^{\text{WP}} \rangle = 0$ (Eq. (A16)). The noise, however, entails that a larger number of perturbations needs to be tried to obtain a faithful gradient estimate and thus a good weight update.

In NP, the error change is given by

$$\Delta E^{\text{NP}}_{\text{pert}} = \overbrace{\tfrac{1}{T}\,\text{tr}[(Wr - d)(\xi^{\text{NP}})^T]}^{\Delta E^{\text{lin,NP}}_{\text{pert}}} + \overbrace{\tfrac{1}{2T}\,\text{tr}[\xi^{\text{NP}} (\xi^{\text{NP}})^T]}^{\equiv \Delta E^{\text{quad,NP}}_{\text{pert}}} \tag{S5}$$

(Eq. (B18)). The linear term is the projection of the node (i.e. output) perturbation onto the output gradient. As discussed in Sec. "Realizable and unrealizable outputs", this gradient is composed of two orthogonal parts, laying in the realizable and in the

unrealizable subspace,

$$\frac{\partial E}{\partial z} = \underbrace{\frac{1}{T} W r}\quad \underbrace{-\frac{d}{T}} \tag{S6}$$

$$= \left[\frac{\partial E}{\partial z}\right]^{\text{real}} + \left[\frac{\partial E}{\partial z}\right]^{\text{unr}}. \tag{S7}$$

$\Delta E_{\text{pert}}^{\text{lin,NP}}$ thus consists of two components, resulting from the projection of ξ^{NP} onto the gradient parts in the realizable and in the unrealizable output subspace,

$$\Delta E_{\text{pert}}^{\text{lin}} = \Delta E_{\text{pert}}^{\text{lin,real}} + \Delta E_{\text{pert}}^{\text{lin,unr}}, \tag{S8}$$

where

$$\Delta E_{\text{pert}}^{\text{lin,real}} \overset{\text{NP}}{=} \frac{1}{T} \sum_{m=1}^{M} \sum_{t=1}^{T} \sum_{j=1}^{N} W_{mj} r_{jt} \xi_{mt}^{\text{NP}} = \sum_{m=1}^{M} \sum_{t=1}^{T} \left[\frac{\partial E}{\partial z}\right]_{mt}^{\text{real}} \xi_{mt}^{\text{NP}}, \tag{S9}$$

$$\Delta E_{\text{pert}}^{\text{lin,unr}} \overset{\text{NP}}{=} -\frac{1}{T} \sum_{m=1}^{M} \sum_{t=1}^{T} d_{mt} \xi_{mt}^{\text{NP}} = \sum_{m=1}^{M} \sum_{t=1}^{T} \left[\frac{\partial E}{\partial z}\right]_{mt}^{\text{unr}} \xi_{mt}^{\text{NP}}. \tag{S10}$$

$\Delta E_{\text{pert}}^{\text{lin,unr}}$ only generates reward noise, since it is uncorrelated with the projection of the noise onto the realizable part of the gradient (which is used for learning), $\langle \Delta E_{\text{pert}}^{\text{lin,real}} \Delta E_{\text{pert}}^{\text{lin,unr}} \rangle = 0$. $\Delta E_{\text{pert}}^{\text{lin,unr}}$ increases the number of perturbations that are necessary to obtain a good weight update from averaging over them, but does not bias the resulting update (Eq. (A17)). Since $\Delta E_{\text{pert}}^{\text{lin,unr}}$ is linear in the perturbations, its effect is independent of the perturbation size (due to our division by σ_{NP} in the NP update equation Eq. (6)). For WP $\Delta E_{\text{pert}}^{\text{lin,unr}}$ is zero. The quadratic part of $\Delta E_{\text{pert}}^{\text{NP}}$ does not contain information on the gradient either and is therefore only a source of reward noise, like the quadratic part of the error change in WP.

Only a fraction of N_{eff}/T of NP's node perturbation strength (as measured by the perturbation variance), lies in the subspace of realizable outputs and can couple to the realizable part of the gradient. Thus for NP the variance of $\Delta E_{\text{pert}}^{\text{lin,real}}$ is smaller than for WP by a factor of N_{eff}/T,

$$\begin{aligned}
\langle\langle \Delta E_{\text{pert}}^{\text{lin,WP}} \rangle\rangle &= \left\langle \left(\sum_{i=1}^{M} \sum_{j=1}^{N} \frac{\partial E}{\partial w_{ij}} \xi_{ij}^{\text{WP}} \right)^2 \right\rangle \\
&= \sum_{im=1}^{M} \sum_{jklp=1}^{N} W_{il} S_{lj} W_{mp} S_{pk} \langle \xi_{ij}^{\text{WP}} \xi_{mk}^{\text{WP}} \rangle \\
&= \sigma_{\text{WP}}^2 \sum_{i=1}^{M} \sum_{jlp=1}^{N} W_{il} S_{lj} S_{pj} W_{ip} \\
&= \sigma_{\text{eff}}^2 (\alpha^2 N_{\text{eff}})^{-1} \operatorname{tr}[W S^2 W^T] \\
&= 2 \frac{\sigma_{\text{eff}}^2}{N_{\text{eff}}} (E - E_{\text{opt}}),
\end{aligned}$$

$$\begin{aligned}
\langle\langle \Delta E_{\text{pert}}^{\text{lin,real,NP}} \rangle\rangle &= \left\langle \left(\sum_{i=1}^{M} \sum_{t=1}^{T} \left[\frac{\partial E}{\partial z}\right]_{it}^{\text{real}} \xi_{it}^{\text{NP}} \right)^2 \right\rangle \\
&= \sum_{im=1}^{M} \sum_{jk=1}^{N} \sum_{ts}^{T} \frac{1}{T^2} W_{ij} r_{jt} W_{mk} r_{ks} \langle \xi_{it}^{\text{NP}} \xi_{ms}^{\text{NP}} \rangle \\
&= \sigma_{\text{eff}}^2 \sum_{i=1}^{M} \sum_{jk=1}^{N} \frac{1}{T} W_{ij} \sum_{t}^{T} \frac{1}{T} r_{jt} r_{kt} W_{ik} \\
&= \sigma_{\text{eff}}^2 T^{-1} \operatorname{tr}[W S W^T] \\
&= 2 \frac{\sigma_{\text{eff}}^2}{T} (E - E_{\text{opt}}) = \frac{N_{\text{eff}}}{T} \cdot \langle\langle \Delta E_{\text{pert}}^{\text{lin,WP}} \rangle\rangle.
\end{aligned} \tag{S11}$$

This leads to a smaller "signal-to-noise ratio" in NP, when measuring the projection of the perturbations onto the gradient using $\Delta E_{\text{pert}}^{\text{lin,real}}$. NP has to compensate this by amplifying the learning signal in $\Delta E_{\text{pert}}^{\text{lin,real}}$ more strongly by a factor of $\sqrt{T/N_{\text{eff}}}$ to achieve the same mean update $\langle w \rangle$ as WP (Eqs. (A2,A4)). Since the learning signal and the related weight update cannot be selectively increased, the entire weight update is larger. This amplification becomes apparent when comparing the size (as measured by the standard deviation, since the mean is zero) of the different factors that ΔE_{pert} is multiplied with in the update rules Eqs. (3,6),

$$\xi_{ij}^{\text{WP}} / \sigma_{\text{WP}}^2 \sim \sigma_{\text{eff}}^{-1} \sqrt{\alpha^2 N_{\text{eff}}}, \qquad \sum_{t=1}^{T} \xi_{it}^{\text{NP}} r_{jt} / \sigma_{\text{eff}}^2 \sim \sigma_{\text{eff}}^{-1} \sqrt{\alpha^2 T} \tag{S12}$$

Here we used Eq. (A22) and assumed for NP that input j is a relevant input, which has strength α^2 (App. A, Sec. "Task setting"). For an irrelevant input, $r_j t = 0$ and ΔE_{pert} is multiplied with zero. Sec. "Strength of weight update fluctuations due to reward noise" explains how the larger weight update leads for NP to larger update fluctuations due to reward noise.

For both WP and NP, the quadratic contribution to the error change equals a normalized sum over the squared output perturbations δz_{it},

$$\Delta E_{\text{pert}}^{\text{quad}} = \frac{1}{2T} \sum_{i=1}^{M} \sum_{t=1}^{T} \delta z_{it}^2, \tag{S13}$$

due to Eq. (S3), Eqs. (A7,A18) for WP and Eqs. (S5,A19) for NP. Since the output perturbations produced by WP and NP have the same summed variance (Eqs. (A20-A22)), also the quadratic contributions to the error have to leading order the same size, which is given by the non-vanishing mean $\langle \Delta E_{\text{pert}}^{\text{quad}} \rangle$,

$$\langle \Delta E_{\text{pert}}^{\text{quad}} \rangle = \frac{1}{2T} \sum_{i=1}^{M} \sum_{t=1}^{T} \langle \delta z_{it}^2 \rangle = \tfrac{1}{2} M \sigma_{\text{eff}}^2, \tag{S14}$$

$$\begin{aligned} \langle\langle \Delta E_{\text{pert}}^{\text{quad}} \rangle\rangle &\overset{\text{WP}}{=} \frac{1}{4T^2} \sum_{im=1}^{M} \sum_{ts=1}^{T} \sum_{jkpq=1}^{N} T^2 S_{jk} S_{pq} \langle \xi_{ij}^{\text{WP}} \xi_{ik}^{\text{WP}} \xi_{mp}^{\text{WP}} \xi_{mq}^{\text{WP}} \rangle - \langle \Delta E_{\text{pert}}^{\text{quad}} \rangle^2 \\ &= \frac{1}{4} \sigma_{\text{WP}}^4 \sum_{im=1}^{M} \sum_{ts=1}^{T} \sum_{jkpq=1}^{N} S_{jk} S_{pq} \left(\delta_{jk} \delta_{pq} + \delta_{im} (\delta_{jp} \delta_{kq} + \delta_{jq} \delta_{kp}) \right) - \frac{1}{4} M^2 \sigma_{\text{eff}}^4 \\ &= \frac{1}{4} \sigma_{\text{WP}}^4 \left(M^2 \operatorname{tr}[S]^2 + 2M \operatorname{tr}[S^2] \right) - \frac{1}{4} M^2 \sigma_{\text{eff}}^4 = \frac{1}{2} \sigma_{\text{eff}}^4 \frac{M}{N_{\text{eff}}}, \end{aligned} \tag{S15}$$

$$\begin{aligned} \langle\langle \Delta E_{\text{pert}}^{\text{quad}} \rangle\rangle &\overset{\text{NP}}{=} \frac{1}{4T^2} \sum_{im=1}^{M} \sum_{ts=1}^{T} \langle \xi_{it}^{\text{NP}} \xi_{it}^{\text{NP}} \xi_{ms}^{\text{NP}} \xi_{ms}^{\text{NP}} \rangle - \langle \Delta E_{\text{pert}}^{\text{quad}} \rangle^2 \\ &= \frac{1}{4T^2} \sigma_{\text{eff}}^4 \sum_{im=1}^{M} \sum_{ts=1}^{T} \left(1 + 2\delta_{im} \delta_{ts} \right) - \frac{1}{4} M^2 \sigma_{\text{eff}}^4 = \frac{1}{2} \sigma_{\text{eff}}^4 \frac{M}{T}. \end{aligned} \tag{S16}$$

To leading order, the size of $\Delta E_{\text{pert}}^{\text{quad}}$ scales with the perturbation strength σ_{eff}^2 and with M because the MT-dimensional output is perturbed with a per-dimension variance of σ_{eff}^2 and the definition Eq. (8) of the error contains a factor of $1/T$ that cancels the T-dependence.

Contributions to the weight update: gradient following, credit assignment-related noise and reward noise

The mean updates of WP and NP align with the gradient and are equal to those of GD (Eqs. (A16,A17, B1)). The updates of WP and NP, however, fluctuate. This slows learning down and, if the perturbations ξ are finite, also limits the final performance. In this section we show that there are two sources of update fluctuations: a credit assignment problem and reward noise. Their impact will be quantified in the subsequent two sections. Thereafter we describe how they influence the different aspects of the error evolution.

Inserting the different contributions to the error change, $\Delta E_{\text{pert}}^{\text{lin,real}}$, $\Delta E_{\text{pert}}^{\text{lin,unr}}$ and $\Delta E_{\text{pert}}^{\text{quad}}$, into the update equations Eqs. (3) and (6), allows us to split the updates into different components,

$$\Delta w_{ij}^{\text{WP}} = -\frac{\eta}{\sigma_{\text{WP}}^2} (\overbrace{\Delta E_{\text{pert}}^{\text{lin,real}} \xi_{ij}^{\text{WP}}}^{\text{I+II}} + \overbrace{\Delta E_{\text{pert}}^{\text{quad}} \xi_{ij}^{\text{WP}}}^{\text{IV}}), \tag{S17}$$

$$\Delta w_{ij}^{\text{NP}} = -\frac{\eta}{\sigma_{\text{NP}}^2} \Big(\underbrace{\Delta E_{\text{pert}}^{\text{lin,real}} \sum_{t=1}^{T} \xi_{it}^{\text{NP}} r_{jt}}_{\text{I+II}} + \underbrace{\Delta E_{\text{pert}}^{\text{lin,unr}} \sum_{t=1}^{T} \xi_{it}^{\text{NP}} r_{jt}}_{\text{III}} + \underbrace{\Delta E_{\text{pert}}^{\text{quad}} \sum_{t=1}^{T} \xi_{it}^{\text{NP}} r_{jt}}_{\text{IV}} \Big), \tag{S18}$$

which may be written as

$$\Delta w_{ij} = \langle \Delta w_{ij} \rangle \qquad \text{(I, from } \Delta E_{\text{pert}}^{\text{lin,real}} \text{ - mean update)} \tag{S19}$$

$$+ \delta w_{ij}^{\text{cr.as}} \qquad \text{(II, from } \Delta E_{\text{pert}}^{\text{lin,real}} \text{ - fluctuations due to credit assignment problem)} \tag{S20}$$

$$+ \delta w_{ij}^{\text{rew.noise,lin}} \qquad \text{(III, from } \Delta E_{\text{pert}}^{\text{lin,unr}} \text{ - fluctuations due to linear reward noise)} \tag{S21}$$

$$+ \delta w_{ij}^{\text{rew.noise,quad}} \qquad \text{(IV, from } \Delta E_{\text{pert}}^{\text{quad}} \text{ - fluctuations due to quadratic reward noise).} \tag{S22}$$

The mean update is proportional to the gradient (Eqs. (A16,A17)),

$$\langle \Delta w_{ij} \rangle = -\eta \frac{\partial E}{\partial w_{ij}}. \tag{S23}$$

The first part of the fluctuations explicitly reads (using Eqs. (S17,S17,S4,S9,S23))

$$\delta w_{ij}^{\text{cr.as}} \overset{\text{WP}}{=} -\frac{\eta}{\sigma_{\text{WP}}^2} \sum_{m=1}^{M} \sum_{k=1}^{N} \frac{\partial E}{\partial w_{mk}} \xi_{mk}^{\text{WP}} \cdot \xi_{ij}^{\text{WP}} + \eta \frac{\partial E}{\partial w_{ij}}, \tag{S24}$$

$$\delta w_{ij}^{\text{cr.as}} \overset{\text{NP}}{=} -\frac{\eta}{\sigma_{\text{NP}}^2} \sum_{m=1}^{M} \sum_{s=1}^{T} \left[\frac{\partial E}{\partial z} \right]_{ms}^{\text{real}} \xi_{ms}^{\text{NP}} \cdot \sum_{t=1}^{T} \xi_{it}^{\text{NP}} r_{jt} + \eta \frac{\partial E}{\partial w_{ij}} \tag{S25}$$

It can be attributed to the *credit assignment* problem of finding, out of the MN weights or MT outputs, the single gradient-parallel component of the perturbation that was responsible for causing $\Delta E_{\text{pert}}^{\text{lin,real}}$, i.e. the linear, instructive part of the error signal. More specifically: Because from the scalar error signal alone it is impossible for WP to determine which of the MN sampled directions was the one parallel to the gradient, WP has to amplify the perturbations ξ_{ij}^{WP} of all weights equally during the constructions of their updates (Eqs. (3,S24)). This implies that all weights fluctuate. The single backpropagation step of NP, reflected in its use of eligibility traces (Eqs. (6,S25)), allows it to solve the credit assignment problem at least partially. Therefore for NP only the MN_{eff} relevant weights are updated. The convergence speed is, however, only limited by the update noise of relevant weights, which is the same for NP and WP, see Eqs. (S32,S33) and compare the identical convergence factors of WP and NP, Eq. (B39).

The second kind of update fluctuations, $\delta w_{ij}^{\text{rew.noise,lin}}$ and $\delta w_{ij}^{\text{rew.noise,quad}}$, read (Eqs. (S21,S10))

$$\delta w_{ij}^{\text{rew.noise,lin}} \overset{\text{WP}}{=} 0, \qquad \delta w_{ij}^{\text{rew.noise,lin}} \overset{\text{NP}}{=} \frac{\eta}{\sigma_{\text{NP}}^2} \sum_{m=1}^{M} \sum_{s=1}^{T} \frac{1}{T} d_{ms} \xi_{mt}^{\text{NP}} \cdot \sum_{t=1}^{T} \xi_{it}^{\text{NP}} r_{jt} \tag{S26}$$

and (Eqs. (S22,S18,S13))

$$\delta w_{ij}^{\text{rew.noise,quad}} \overset{\text{WP}}{=} -\frac{\eta}{\sigma_{\text{WP}}^2} \frac{1}{2T} \sum_{m=1}^{M} \sum_{s=1}^{T} \left(\sum_{k=1}^{N} \xi_{mk}^{\text{WP}} r_{ks} \right)^2 \cdot \xi_{ij}^{\text{WP}}, \tag{S27}$$

$$\delta w_{ij}^{\text{rew.noise,quad}} \overset{\text{NP}}{=} -\frac{\eta}{\sigma_{\text{NP}}^2} \frac{1}{2T} \sum_{m=1}^{M} \sum_{s=1}^{T} \left(\xi_{ms}^{\text{NP}} \right)^2 \cdot \sum_{t=1}^{T} \xi_{it}^{\text{NP}} r_{jt}. \tag{S28}$$

They are caused by reward noise ($\Delta E_{\text{pert}}^{\text{lin,unr}}$ and $\Delta E_{\text{pert}}^{\text{quad}}$, respectively). Multiplying the reward noise in $E^{\text{pert}} - E$ with the applied perturbations or eligibility traces in the construction of updates, Eqs. (3,6), results in random update contributions that are unrelated to the gradient (see Eqs. (S27,S28) in contrast to Eqs. (S24,S25))). Because the reward noise is independent of the weight mismatch, the amplitudes of these fluctuations do not change over the course of training, which prevents learning with arbitrary precision. In contrast, the informative $\Delta E_{\text{pert}}^{\text{lin,real}}$ diminishes with training.

Strength of credit assignment-related weight update fluctuations and the dimensionality argument for our task

This section computes and compares the fluctuation strengths due to the credit-assignment problem in WP and NP learning. Thereafter it quantitatively states a dimensionality argument, which adapts the one that is commonly used to compare WP, NP and GD learning (main text, introduction) to our type of task.

The strength of the update noise due to the credit assignment problem, i.e. the variance of $\delta w_{ij}^{\text{cr.as}}$, depends only on the contribution $\Delta E_{\text{pert}}^{\text{lin,real}}$ of the error change (Line (S20)) and is thus independent of d and σ_{eff} (Eqs. (S17,S18,S3,S9)). For WP it is

$$\begin{aligned}
\langle\langle \delta w_{ij}^{\text{cr.as}} \rangle\rangle &\overset{\text{WP}}{=} \left\langle \left(-\tfrac{\eta}{\sigma_{\text{WP}}^2} \Delta E_{\text{pert}}^{\text{lin}} \xi_{ij}^{\text{WP}} \right)^2 \right\rangle - \langle \Delta w_{ij} \rangle^2 \\
&= \tfrac{\eta^2}{\sigma_{\text{WP}}^4} \sum_{mn=1}^{M} \sum_{kl=1}^{N} \frac{\partial E}{\partial w_{mk}} \frac{\partial E}{\partial w_{nl}} \underbrace{\left\langle \xi_{mk}^{\text{WP}} \xi_{nl}^{\text{WP}} \xi_{ij}^{\text{WP}} \xi_{ij}^{\text{WP}} \right\rangle}_{=\sigma_{\text{WP}}^4 \left(\delta_{mn}\delta_{kl} + 2\delta_{mni}\delta_{jkl} \right)} - \eta^2 \left(\frac{\partial E}{\partial w_{ij}} \right)^2 \\
&= \eta^2 \left| \frac{\partial E}{\partial w} \right|^2 + \eta^2 \left(\frac{\partial E}{\partial w_{ij}} \right)^2 .
\end{aligned} \tag{S29}$$

These credit assignment-related random fluctuations in the update of w_{ij} thus grow quadratically with the overall length of the error gradient and with the size of its component in w_{ij}-direction. The first dependency arises because any weight perturbation is multiplied with the global error change, which is in linear order proportional to the gradient length. The second dependency arises

because weight changes parallel to the gradient fluctuate twice as strongly as in perpendicular directions due to their correlation with error changes.

For NP, the credit assignment-dependent weight update noise strength is

$$\begin{aligned}\langle\langle \delta w_{ij}^{\text{cr.as}}\rangle\rangle &\overset{\text{NP}}{=} \left\langle \left(-\frac{\eta}{\sigma_{\text{NP}}^2} \Delta E_{\text{pert}}^{\text{lin,real}} \sum_{t=1}^{T} \xi_{it}^{\text{NP}} r_{jt} \right)^2 \right\rangle - |\langle \Delta w_{ij}\rangle|^2 \\ &= \frac{\eta^2}{\sigma_{\text{NP}}^4} \sum_{mn=1}^{M} \sum_{stuv=1}^{T} \left[\frac{\partial E}{\partial z}\right]_{ms}^{\text{real}} \left[\frac{\partial E}{\partial z}\right]_{nt}^{\text{real}} \underbrace{\left\langle \xi_{ms}^{\text{NP}} \xi_{nt}^{\text{NP}} \xi_{iu}^{\text{NP}} \xi_{iv}^{\text{NP}} \right\rangle}_{=\sigma_{\text{NP}}^4 \left(\delta_{mn}\delta_{st}\delta_{uv} + \delta_{imn}(\delta_{su}\delta_{tv} + \delta_{sv}\delta_{tu})\right)} r_{ju} r_{jv} - \eta^2 \left(\frac{\partial E}{\partial w_{ij}}\right)^2 \\ &= \eta^2 \left| \left[\frac{\partial E}{\partial z}\right]^{\text{real}} \right|^2 T S_{jj} + \eta^2 \left(\frac{\partial E}{\partial w_{ij}}\right)^2 . \end{aligned} \tag{S30}$$

The first term is proportional to the squared length of the realizable part of the output gradient and to the strength of the jth input. This is because in the weight update rule node perturbations are multiplied with the global error change (which is in turn proportional to the output gradient length) and with the jth input. The second term arises again because weight changes in the direction of the gradient fluctuate twice as strongly due to their correlation with the error change. The first term is generally different from that of WP, which can cause differences in convergence speeds - compare SM4 and main text, Sec. "Reservoir computing-based drawing task". For the case of equally strong latent inputs, which we focus on in our analytical computations, however, noise variances are equal for WP and NP: in the rotated space of input components $r_{\mu t}$ introduced in Sec. "Task relevant and irrelevant weights", the squared norm of the weight error gradient is

$$\begin{aligned}\left|\frac{\partial E}{\partial w}\right|^2 &= \sum_{m=1}^{M} \sum_{\mu=1}^{N} \left(\frac{\partial E}{\partial w_{m\mu}}\right)^2 = \sum_{m=1}^{M} \sum_{\mu=1}^{N} \left(\sum_{t=1}^{T} \frac{\partial E}{\partial z_{mt}} r_{\mu t} \right)^2 \\ &= \sum_{m=1}^{M} \sum_{\mu=1}^{N_{\text{eff}}} \left(\frac{\partial E}{\partial z_m}, \hat{r}_\mu\right)^2 T\alpha^2 \overset{(*)}{=} \left| \left[\frac{\partial E}{\partial z}\right]^{\text{real}} \right|^2 T\alpha^2 . \end{aligned} \tag{S31}$$

Here we introduced the temporal unit vectors $\hat{r}_\mu = \frac{1}{\sqrt{T}\alpha} r_\mu$ and used in $(*)$ that the $\hat{r}_\mu$ with $\mu = 1, ..., N_{\text{eff}}$ form a basis for the subspace in which the realizable part of the node gradient lies. In the space of rotated inputs we thus have

$$\langle\langle \delta w_{i\mu}^{\text{cr.as}}\rangle\rangle \overset{\text{WP}}{=} \eta^2 \left|\frac{\partial E}{\partial w}\right|^2 + \eta^2 \left(\frac{\partial E}{\partial w_{i\mu}}\right)^2 \qquad \text{for every input } \mu, \tag{S32}$$

$$\langle\langle \delta w_{i\mu}^{\text{cr.as}}\rangle\rangle \overset{\text{NP}}{=} \begin{cases} \eta^2 \left|\frac{\partial E}{\partial w}\right|^2 + \eta^2 \left(\frac{\partial E}{\partial w_{i\mu}}\right)^2 & \text{for relevant inputs } \mu = 1, \cdots, N_{\text{eff}}, \\ 0 & \text{for irrelevant inputs } \mu = N_{\text{eff}} + 1, \cdots, N. \end{cases} \tag{S33}$$

Eq. (S32) reduces for an irrelevant input μ to $\langle\langle \delta w_{i\mu}^{\text{cr.as}}\rangle\rangle \overset{\text{WP}}{=} \eta^2 \left|\frac{\partial E}{\partial w}\right|^2$, i.e. for WP there are credit assignment-related update fluctuations also in irrelevant weight space directions, in contrast to NP. Eqs. (S32,S33) imply that the average change of a single relevant weight due to credit assignment noise, $\sqrt{\langle\langle \delta w_{i\mu}^{\text{cr.as}}\rangle\rangle}$, is at least as large as the size of the deterministic improvement of the entire weight vector along the gradient, $\eta \left|\frac{\partial E}{\partial w}\right| = |\langle \Delta w\rangle|$. The average size of the entire weight vector change due to credit assignment noise can be computed by summing over Eqs. (S32,S33): Using $\sum_{i,\mu} \eta^2 \left(\frac{\partial E}{\partial w_{i\mu}}\right)^2 = \eta^2 \left|\frac{\partial E}{\partial w}\right|^2 = |\langle \Delta w\rangle|^2$ yields

$$\langle |\delta w^{\text{cr.as}}|^2\rangle = \begin{cases} (MN + 1) \cdot |\langle \Delta w\rangle|^2 & \text{for WP}, \\ (MN_{\text{eff}} + 1) \cdot |\langle \Delta w\rangle|^2 & \text{for NP}. \end{cases} \tag{S34}$$

This means that the weight change due to credit assignment noise is much larger than that due to the deterministic update $\langle \Delta w\rangle$. The contributions of noise and deterministic gradient following simply add up to the total average square weight change,

$$\langle |\Delta w|^2\rangle = |\langle \Delta w\rangle|^2 + \langle |\delta w^{\text{cr.as}}|^2\rangle, \tag{S35}$$

in absence of reward noise. Since update noise will influence and often increase the error in a nonlinear way (Eq. (8)), it might seem as if learning is impossible. However, for sufficiently small updates also with a nonlinear error function the deterministic update parts add up, as they always point into approximately the same direction, while the fluctuations partly cancel each other. For

infinitesimally small update size, after n updates, the mean weight change is $n \cdot \langle \Delta w \rangle$, while the standard deviation of the summed fluctuations of a single weight scales only as $\sqrt{n} \cdot \sqrt{\langle |\delta w^{\text{cr.as}}|^2 \rangle}$. This way WP and NP can still learn by adopting a smaller learning rate and averaging out fluctuations over more updates.

Eq. (S34) seems to furthermore imply that NP is much more efficient in learning than WP, as its noise is much smaller. This is the classical dimensionality argument cited in the Introduction section of the main text. However, for a single input-output learning task noise-related changes of only the MN_{eff} task relevant weights influence the error. Thus, the relevant amount of credit assignment noise is actually the same for WP and NP, leading to the same maximal speed of convergence (Fig. 1).

Strength of weight update fluctuations due to reward noise

Here we give the scaling of reward noise-related update fluctuations and explain why these are larger for NP than for WP, which will explain the larger final error of NP.

The scaling of $\delta w_{ij}^{\text{rew.noise,lin}}$ for NP follows from Eqs. (S21,S18,S10). Eq. (S10) yields $\Delta E_{\text{pert}}^{\text{lin,unr}} \overset{\text{NP}}{\sim} \sigma_{\text{eff}} \alpha_d \sqrt{\frac{M}{T}}$, since the "size" of the sum of centered noise terms scales with the square root of the number of summands. We here consider as the size of $\delta w_{ij}^{\text{rew.noise,lin}}$ the standard deviation of the sum (Eq. (S18), term III), as its average vanishes. The formal reason that this common approach works is that we compute the strengths (variances) of noise terms $\delta w_{ij}^{\text{rew.noise,lin}}$ from a product of two independent random variables $X = \Delta E_{\text{pert}}^{\text{lin,unr}}$ and $Y = \sum_{t=1}^{T} \xi_{it}^{\text{NP}} r_{jt}$, which both have zero mean. The well known general formula $\text{Var}(XY) = \text{E}(X)^2\text{Var}(Y) + \text{E}(Y)^2\text{Var}(X) + \text{Var}(X)\text{Var}(Y)$ for independent random variables X, Y, thus tells us that the leading order scaling in each random variable arises from its squared average or from its variance, i.e. its squared standard deviation. For our computations we therefore consider as characteristic size of a term its average or its standard deviation, depending on which has the leading order scaling, and use its square to compute the final noise variance scaling. Thus the variance of $\delta w^{\text{rew.noise,lin}}$ can be obtained by simply multiplying the variances of $\Delta E_{\text{pert}}^{\text{lin,unr}}$ and $\sum_{t=1}^{T} \xi_{it}^{\text{NP}} r_{jt}$,

$$\langle\langle \delta w_{ij}^{\text{rew.noise,lin}} \rangle\rangle \overset{\text{WP}}{=} 0, \tag{S36}$$

$$\begin{aligned} \langle\langle \delta w_{ij}^{\text{rew.noise,lin}} \rangle\rangle &\overset{\text{NP}}{=} \frac{\eta^2}{\sigma_{\text{eff}}^4} \cdot \langle\langle \Delta E_{\text{pert}}^{\text{lin,unr}} \rangle\rangle \cdot \Big\langle\Big\langle \sum_{t=1}^{T} \xi_{it}^{\text{NP}} r_{jt} \Big\rangle\Big\rangle \\ &= \eta^2 \alpha_d^2 \alpha^2 M = 2\eta^2 \alpha^2 E_{\text{opt}} \qquad \text{for relevant weights only.} \end{aligned} \tag{S37}$$

(We have thus obtained the size of the product from the product of sizes, as for deterministic quantities.) We conclude that the size of $\delta w^{\text{rew.noise,lin}}$ is 0 for WP and scales for NP as

$$\delta w_{ij}^{\text{rew.noise,lin}} \overset{\text{NP}}{\sim} \eta \alpha_d \sqrt{\alpha^2 M} \qquad \text{for relevant weights only.} \tag{S38}$$

The scaling of $\delta w_{ij}^{\text{rew.noise,quad}}$ can be computed likewise: $\Delta E_{\text{pert}}^{\text{quad}}$ is a sum of independent positive random variables (Eq. (S13)) such that its size has a contribution from the summed means (Eq. (S14)), which scales with M, and a contribution from the summed fluctuations (Eqs. (S15,S16)), which scales with $\sqrt{M}$. As ξ_{ij}^{WP} and $\sum_{t=1}^{T} \xi_{it}^{\text{NP}} r_{jt}$ have zero mean, we obtain the full expressions

$$\begin{aligned} \langle\langle \delta w_{ij}^{\text{rew.noise,quad}} \rangle\rangle &\overset{\text{WP}}{=} \frac{\eta^2}{\sigma_{\text{WP}}^4} \cdot \Big(\langle \Delta E_{\text{pert}}^{\text{quad}} \rangle^2 + \langle\langle \Delta E_{\text{pert}}^{\text{quad}} \rangle\rangle \Big) \cdot \langle\langle \xi_{ij}^{\text{WP}} \rangle\rangle \\ &= \frac{\eta^2}{\sigma_{\text{WP}}^2} \cdot \Big(\frac{1}{4} M^2 \sigma_{\text{eff}}^4 + \frac{1}{2} \sigma_{\text{eff}}^4 \frac{M}{N_{\text{eff}}} \Big) = \frac{1}{4} \eta^2 \sigma_{\text{eff}}^2 \alpha^2 \big(M^2 N_{\text{eff}} + 2M \big) \end{aligned} \tag{S39}$$

$$\begin{aligned} \langle\langle \delta w_{ij}^{\text{rew.noise,quad}} \rangle\rangle &\overset{\text{NP}}{=} \frac{\eta^2}{\sigma_{\text{eff}}^4} \cdot \Big(\langle \Delta E_{\text{pert}}^{\text{quad}} \rangle^2 + \langle\langle \Delta E_{\text{pert}}^{\text{quad}} \rangle\rangle \Big) \cdot \langle\langle \sum_{t=1}^{T} \xi_{it}^{\text{NP}} r_{jt} \rangle\rangle \\ &= \frac{\eta^2}{\sigma_{\text{eff}}^4} \cdot \Big(\frac{1}{4} M^2 \sigma_{\text{eff}}^4 + \frac{1}{2} \sigma_{\text{eff}}^4 \frac{M}{T} \Big) \cdot \sigma_{\text{eff}}^2 \sum_{t=1}^{T} r_{jt}^2 \\ &= \begin{cases} \frac{1}{4} \eta^2 \sigma_{\text{eff}}^2 \alpha^2 \big(M^2 T + 2M \big) & \text{for relevant weights} \\ 0 & \text{for irrelevant weights} \end{cases} \end{aligned} \tag{S40}$$

for the noise variances. To leading order, the size of update fluctuations induced by quadratic reward noise is thus

$$\delta w_{ij}^{\text{rew.noise,quad}} \overset{\text{WP}}{\approx} \frac{1}{2}\eta\sigma_{\text{eff}}\alpha M\sqrt{N_{\text{eff}}} \qquad \text{for all weights,} \tag{S41}$$

$$\delta w_{ij}^{\text{rew.noise,quad}} \overset{\text{NP}}{\approx} \frac{1}{2}\eta\sigma_{\text{eff}}\alpha M\sqrt{T} \qquad \text{for relevant weights only.} \tag{S42}$$

The different scaling ultimately results from the fact that the output variability of NP in the realizable output subspace is by a factor of N_{eff}/T smaller than for WP (Eq. (S11)). NP has to compensate this by amplifying the learning signal in $\Delta E_{\text{pert}}^{\text{lin,real}}$ more strongly by a factor of $\sqrt{T/N_{\text{eff}}}$ (Eq. (S12)) to achieve the same beneficial mean update $\langle \Delta w \rangle$. It is, however, unavoidable to apply the stronger amplification to the entire ΔE_{pert}. Because its part $\Delta E_{\text{pert}}^{\text{quad}}$ is – in contrast to $\Delta E_{\text{pert}}^{\text{lin,real}}$ – to leading order the same for WP and NP (Eq. (S14)), the stronger overall amplification leads to larger fluctuations $\delta w_{\text{NP}}^{\text{rew.noise,quad}} \approx \sqrt{T/N_{\text{eff}}} \cdot \delta w_{\text{WP}}^{\text{rew.noise,quad}}$ in NP.

Components of the recurrence relation

Here we leverage the concepts developed in the preceding sections to explain the origin and scaling of each component of the recurrence relation. Together with the next section, in which the recurrence relation is solved and the error dynamics characterized, this provides a rigorous understanding of how different task properties affect specific aspects of the error evolution.

Distinguishing the contributions $\langle \Delta w_{ij} \rangle$, $\delta w_{ij}^{\text{cr.as}}$, $\delta w_{ij}^{\text{rew.noise,lin}}$ and $\delta w_{ij}^{\text{rew.noise,quad}}$ (Lines (S19-S22)) to the updates Δw allows to examine their different effects on the evolution of the expected error. The fluctuations are independent of each other and have zero expectation value. The expected error after an update has the following components:

$$\langle E(n) \rangle = \tfrac{1}{2} \langle \text{tr}[(W(n-1) + \Delta w) S (W(n-1) + \Delta w)^T] \rangle + E_{\text{opt}} \tag{S43}$$

$$= \langle E(n-1) \rangle \qquad \text{(error before update)} \tag{S44}$$

$$+ \text{tr}[W S \langle \Delta w \rangle^T] + \tfrac{1}{2} \text{tr}[\langle \Delta w \rangle S \langle \Delta w \rangle^T] \qquad \text{(mean updates follow gradient as for GD)} \tag{S45}$$

$$+ \tfrac{1}{2} \langle \text{tr}[\delta w^{\text{cr.as}} S (\delta w^{\text{cr.as}})^T] \rangle \qquad \text{(update noise from credit assignment)} \tag{S46}$$

$$+ \tfrac{1}{2} \langle \text{tr}[\delta w^{\text{rew.noise,lin}} S (\delta w^{\text{rew.noise,lin}})^T] \rangle \qquad \text{(update noise from linear reward noise)} \tag{S47}$$

$$+ \tfrac{1}{2} \langle \text{tr}[\delta w^{\text{rew.noise,quad}} S (\delta w^{\text{rew.noise,quad}})^T] \rangle \qquad \text{(update noise from quadratic reward noise).} \tag{S48}$$

Because the mean update is equal to that of GD, the first two contributions to $\langle E(n) \rangle$ would lead to

$$E(n) = (1 - 2\eta\alpha^2 + \eta^2\alpha^4) \cdot \left(E(n-1) - E_{\text{opt}} \right) + E_{\text{opt}} \qquad \text{(for (S44) and (S45)),} \tag{S49}$$

cf. Eqs. (B2) to (B5). $-2\eta\alpha^2$ results from the beneficial term linear in $\langle \Delta w \rangle$, while $\eta^2\alpha^4$ results from the quadratic effect of an update perfectly parallel to the gradient on the error.

The third contribution, (S46), occurs because WP and NP solve the credit assignment problem by random search. The resulting update fluctuations $\delta w^{\text{cr.as}}$ add a diffusive part to the evolution of relevant weights (or, for non-rotated input space: in the relevant weight subspace), which on average leads to an increase in error. Fluctuations of irrelevant weights present in WP (Eq. (S32)) do not contribute; this is reflected by the multiplication of $\delta w^{\text{cr.as}}$ with S. Eq. (S34) shows that (S46) yields a detrimental quadratic term that is $(M N_{\text{eff}} + 1)$ larger than the detrimental quadratic term in (S45). We therefore have

$$E(n) = \underbrace{(1 - 2\eta\alpha^2 + \eta^2\alpha^4 (M N_{\text{eff}} + 2))}_{=a} \cdot \left(E(n-1) - E_{\text{opt}} \right) + E_{\text{opt}} \qquad \text{(for (S44) to (S46)),} \tag{S50}$$

which implicates a $(M N_{\text{eff}} + 2)$ times smaller optimal learning rate than with GD (Eqs. (B45,B5)).

The increase in error resulting from the last two contributions, (S47) and (S48) stems from the update fluctuations $\delta w^{\text{rew.noise}}$ due to reward noise. Because these fluctuations are independent of gradient and weights (Eqs. (S39,S40)), the magnitude of their additive contribution does not change over the course of training. The contribution therefore yields the per-update error increase $b_{\text{WP|NP}}$ of the recurrence relation (Eqs. (8,B37)). b can be split into two parts $b = b^{\text{lin}} + b^{\text{quad}}$ (Eq. (B44)) according to the reward noise source: b^{lin} is only present for NP and arises from the linear reward noise due to unrealizable parts of the target (S47). b^{quad} arises from quadratic reward noise due to the quadratic nonlinearity of the error function (S48). Thus, including all five contributions results in

$$E(n) = (1 - 2\eta\alpha^2 + \eta^2\alpha^4 (M N_{\text{eff}} + 2)) \cdot \left(E(n-1) - E_{\text{opt}} \right) + b^{\text{lin}} + b^{\text{quad}} + E_{\text{opt}} \qquad \text{(for (S44) to (S48)).} \tag{S51}$$

b_{WP}, $b_{\text{NP}}^{\text{lin}}$ and $b_{\text{NP}}^{\text{quad}}$ are given in Eqs. (B42–B44).

Characteristics of the average error dynamics

With all components of the recurrence relation explained and quantified, here we characterize the resulting dynamics of the average error. In particular, we derive the scaling of the final error.

The recurrence relation Eqs. (B37,S51),

$$\big(\langle E(n)\rangle - E_{\text{opt}}\big) = \big(\langle E(n-1)\rangle - E_{\text{opt}}\big)\cdot a + b, \tag{S52}$$

leads, provided that $a < 1$, to an exponential decay (cf. Eq. (B38))

$$\langle E(n)\rangle = \big(E(0) - E_{f,\text{unr}}\big)\cdot a^n + E_{f,\text{unr}} \tag{S53}$$

to a final error

$$E_{f,\text{unr}} = E_f + E_{f,d} + E_{\text{opt}}, \qquad E_f = \frac{b^{\text{quad}}}{1-a}, \qquad E_{f,d} = \frac{b^{\text{lin}}}{1-a}. \tag{S54}$$

$E_{f,\text{unr}}$ is the total final error including contributions due to unrealizable target compoents. E_f captures the final error for the case of realizable targets. For NP unrealizable target components d increase the final error by $E_{f,d}$ (which is zero for WP). Further, unrealizable target components increase the final error by E_{opt}, i.e. by the error that necessarily remains even for optimal weights (Eq. (A15)), because the target is not realizable by the network. Since E_{opt} is known beforehand and for any learning rule represents an inevitable shift in error, it could be absorbed into a redefined E. The final error is then still limited by the accumulation of error due to reward noise, which leads to update noise entering the recurrence relation through b (Eq. (S51) and lines (S47, S48)).

In the beginning of learning, when the weight mismatch is large, following the large gradient leads to fast improvements. As the weights approach their target values, the gradient becomes smaller, but the error contributions that result from reward noise stay constant. Gradient-related improvements and deterioration due to reward noise on average balance if the error E has the size of the final error of the average error dynamics, $E = E_{f,\text{unr}}$. Around this point learning yields hardly any or no improvement. If E happens to become smaller than $E_{f,\text{unr}}$, learning even has a deteriorating effect on average, because of the reward noise.

The deteriorating effect of quadratic reward noise due to finite perturbations is present in both WP and NP and unaffected by unrealizable parts of targets. It may be best illustrated for the case where all relevant weights have already assumed their target values: Then the weight error gradient is zero. Still, a finite random perturbation of the weights or outputs leads with probability one to an increase in error due to the quadratic error function. Therefore the update rules induce a finite weight change in the direction opposite to the perturbation. This prevents the weights from staying at or reaching optimal values and leads to a final error larger than E_{opt}.

We note that the recurrence relation Eq. (B37) may also be understood as a leaky integration of b. For slow convergence, the factor $1-a$ in the denominator (which arises from the discrete updating process) can be interpreted as the leak rate $-\ln(a) \approx 1-a$ of a corresponding continuous exponential decay $\sim \exp(-\ln(a)t)$ (Fig. 1) that equals the actual decay at the points where $t \in \mathbb{N}_0$. In this picture, the contribution $b/(1-a)$ to the final error results from integrating the per-update error increase b over $1/(1-a)$ updates. This determines the error once the contribution due to the initialization, $\big(E(0) - E_{f,\text{unr}}\big)\cdot a^n$, has faded away.

The leading order contributions in M, N_{eff} and T to E_f as given in Eqs. (13),(14) can be computed using Eqs. (B42,B43,B39) or line (S48) and Eqs. (S41,S42,B39) (note that $E_f = \frac{b^{\text{quad}}}{1-a}$ does not incorporate b^{lin} and that line (S48) equals b^{quad}),

$$E_f^{\text{WP}} = \frac{b_{\text{WP}}(\eta)}{1-a(\eta)} \approx \frac{\sigma_{\text{eff}}^2}{8}\cdot \eta^2\alpha^4 \frac{M N_{\text{eff}}}{1-a(\eta)}\cdot M^2 N_{\text{eff}}, \tag{S55}$$

$$E_f^{\text{NP}} = \frac{b_{\text{NP}}^{\text{quad}}(\eta)}{1-a(\eta)} \approx \frac{\sigma_{\text{eff}}^2}{8}\cdot \underbrace{\eta^2\alpha^4 \frac{M N_{\text{eff}}}{1-a(\eta)}}_{\equiv c(\eta)}\cdot M^2 T. \tag{S56}$$

For our discussion, we have split the two results into three corresponding factors: (i) The first factor, $\sigma_{\text{eff}}^2/8$, reflects that $E_f^{\text{WP|NP}}$ will be of significant size only when the noise strength σ_{eff}^2 is sufficiently large. In this case the quadratic reward noise becomes sizeable and corrupts learning. (ii) The second factor contains the dependence of the final error on the learning rate, via a factor that we abbreviate by $c(\eta)$,

$$c(\eta) \equiv \eta^2\alpha^4 \frac{M N_{\text{eff}}}{1-a(\eta)}, \quad c(\eta^*) \approx 1. \tag{S57}$$

Here it is most important that this factor is approximately 1 at the optimal learning rate η^* (Eq. (12)), which we usually consider throughout the paper. $c(\eta)$ goes to zero for small learning rates (Eq. (B39)) and diverges for $\eta \to 2\eta^*$, since then $a \to 1$ (Eq. (B47)). We will furthermore see below that it describes the η-dependence of the ratio between diffusion of irrelevant and improvements of relevant weights (Eq. (S64)), as well as the additional increase in final error of NP caused by unrealizable target components (Eq. (S58)). (iii) The last factor originates from the scaling of the update fluctuations $\delta w^{\text{rew.noise,quad}}$, Eqs. (S41,S42), which enter quadratically. It can be further refactored as M times the effective perturbation dimension MN_{eff} or MT. The latter reflects the fact that WP and NP generate perturbations in spaces of dimensions MN_{eff} and MT, but only the projection onto the output gradient is useful for learning. The other factor M originates from the overall scaling of the error E with M, as the error function Eq. (8) contains a sum over M outputs. In other words, if E was fully normalized, by $1/(MT)$, E_f would scale with the effective perturbation dimension.

The additional contribution to the final error of NP due to the presence of unrealizable target components, $E^{\text{NP}}_{f,d}$ (Eq. (S54)), can be obtained from Eqs. (B43,B39) or line (S47) and Eq. (B39),

$$E^{\text{WP}}_{f,d} = 0, \qquad E^{\text{NP}}_{f,d} = \frac{b^{\text{lin}}}{1-a} = c(\eta) \cdot E_{\text{opt}}, \tag{S58}$$

with the factor $c(\eta)$ from Eq. (S57). Since $c(\eta^*) \approx 1$, learning at η^* means that unrealizable target components increase NP's final error by approximately $2E_{\text{opt}}$ instead of E_{opt} as for WP (Fig. 3). As E_{opt} is independent of σ_{eff}, $E^{\text{NP}}_{f,d}$ remains finite even in the limit of infinitesimally small perturbations where E_f vanishes. For small perturbations, $E^{\text{NP}}_{f,d}$ and the unavoidable E_{opt} can therefore become the dominant contributions to the final error.

SM 2

Analysis of Weight Diffusion

This part provides a quantitative mathematical analysis of the weight changes in irrelevant directions of the weight space.

Weight diffusion due to credit assignment- and reward noise-related update fluctuations

As described in SM1, Sec. "Task relevant and irrelevant weights", we can assume without loss of generality that the first N_{eff} inputs are orthogonal and of strength α^2, while the remaining $N - N_{\text{eff}}$ inputs are zero. The first N_{eff} synaptic weights are thus relevant in the sense that their value influences the output while the remaining ones are irrelevant. How do irrelevant weights change during the modeled learning? For NP the update $\Delta w_{ij}^{\text{NP}}$ is proportional to the eligibility trace $\sum_t \xi_{it}^{\text{NP}} r_{jt}$ (Eq. (6)), which is zero when the jth input is zero, $r_{jt} = 0$, for all t. Therefore NP does not update irrelevant weights. WP, on the other hand, perturbs and updates also the irrelevant weights (Eq. (3)). In the following we therefore focus on WP.

The evolution of the relevant weights is independent of the evolution of the irrelevant weights. This is because only the relevant weights influence the reward and they are perturbed independently of the irrelevant ones. The independence is echoed in the fact that the learning characteristics of WP only depend on N_{eff} but not on N (Eqs. (B39,B42)). The perturbations of the irrelevant weights are also independent of each other, such that it suffices to consider the evolution of a single one. We call it w_{irrel} and its perturbation $\xi_{\text{irrel}}^{\text{WP}}$. The expectation value of its update $\Delta w_{\text{irrel}}^{\text{WP}}$ is zero as $\xi_{\text{irrel}}^{\text{WP}}$ does not influence the reward,

$$\langle \Delta w_{\text{irrel}}^{\text{WP}} \rangle = -\frac{\eta}{\sigma_{\text{WP}}^2} \langle \underbrace{\left(E^{\text{pert}} - E\right)}_{\text{indep. of } \xi_{\text{irrel}}^{\text{WP}}} \xi_{\text{irrel}}^{\text{WP}} \rangle = -\frac{\eta}{\sigma_{\text{WP}}^2} \langle E^{\text{pert}} - E \rangle \langle \xi_{\text{irrel}}^{\text{WP}} \rangle = 0. \tag{S59}$$

This implies that the expectation value of the irrelevant weight at step n taken with respect to the noise applied at all times, $\langle w_{\text{irrel}}(n) \rangle_{\text{all}}$, stays identical to the initial weight,

$$\langle w_{\text{irrel}}(n) \rangle_{\text{all}} = w_{\text{irrel}}(0). \tag{S60}$$

As a consequence the empirical ensemble mean, obtained by averaging over the irrelevant weights at a step n in simulations, does not drift, cf. main text, Fig. 2, and Fig. S1. In contrast, the variance of the irrelevant weight, $\langle\langle w_{\text{irrel}}(n) \rangle\rangle_{\text{all}}$, increases from one update to the next. This is because the variance of a weight update,

$$\begin{aligned} \langle\langle \Delta w_{\text{irrel}} \rangle\rangle &= \frac{\eta^2}{\sigma_{\text{WP}}^4} \left\langle \left(E^{\text{pert}} - E\right)^2 \cdot (\xi_{\text{irrel}}^{\text{WP}})^2 \right\rangle \\ &= 2\eta^2 \alpha^2 (E - E_{\text{opt}}) + \tfrac{1}{4} \eta^2 \sigma_{\text{eff}}^2 \alpha^2 \left(M^2 N_{\text{eff}} + 2M\right), \end{aligned} \tag{S61}$$

(derived below, Eq. (S62)) is nonzero and this variance adds to the variance present before the update. Therefore the empirical standard deviation of the sample of irrelevant weights increases with each learning step n, cf. Fig. 2.

We have seen in SM1, Sec. "Contributions to the weight update: gradient following, credit assignment-related noise and reward noise" that the WP weight update can be split into three parts, originating from gradient following (Line (S19)), the specific way of solving the credit assignment problem (Line (S20)) and reward noise (Line (S22)). Since the variance increase in the irrelevant weight, Eq. (S61), stems from the weight update, we can attribute its different terms to the different weight update contributions. For this we first note that the gradient following contribution line Eq. (S19) does not influence Δw_{irrel}, since it is parallel to the error gradient (Eq. (S23)), which lies in the subspace of relevant weights. Since the variances due to independent noise sources add up, the variance of an irrelevant weight's update (Eq. (S61)) is the sum of the variances of credit assignment-related (Eq. (S32)) and reward noise induced fluctuations (Eq. (S39))

$$\begin{aligned} \langle\langle \Delta w_{\text{irrel}} \rangle\rangle &= \langle\langle \delta w_{\text{irrel}}^{\text{cr.as}} \rangle\rangle + \langle\langle \delta w_{\text{irrel}}^{\text{rew.noise,quad}} \rangle\rangle \\ &= \eta^2 \left| \frac{\partial E}{\partial w} \right|^2 + \frac{1}{4} \eta^2 \sigma_{\text{eff}}^2 \alpha^2 \left(M^2 N_{\text{eff}} + 2M\right). \end{aligned} \tag{S62}$$

Noticing that $\left|\frac{\partial E}{\partial w}\right|^2 = \text{tr}[W S^2 W^T] = 2\alpha^2 (E - E_{\text{opt}})$ yields Eq. (S61).

The derivation shows that the first term in Eq. (S61) originates from credit assignment noise while the second originates from reward noise. Consequently, the first term decreases with the learning progress (as it depends on E) and does not depend on the

perturbation strength σ^2_{WP}. It dominates at the beginning of the training, when the error is large. The second term is, in contrast, constant during learning and depends on the perturbation strength σ^2_{WP}. For finite σ^2_{WP} it becomes influential at the end of training when the error is small. The first term in Eq. (S61) converges for finite σ^2_{WP} to a finite value since the second term implies that the task performance error E never converges to zero (Eq. (B35)). While the relevant weights then do not improve further and just fluctuate around their optimal values, the irrelevant weights keep diffusing (Fig. S1b, left hand side, and Sec. "Weight diffusion for stationary error in presence of reward noise" below). For infinitesimal perturbation strength, the second term in Eq. (S61) is zero and the error E as well as the first term converge to zero. The relevant weights converge and the diffusion of irrelevant weights is only transient (Fig. S1a, right hand side, and the next Sec. "Transient weight diffusion due to credit assignment-related update fluctuations").

Transient weight diffusion due to credit assignment-related update fluctuations

In the following we quantitatively analyze the transient growth of weights for infinitesimal perturbation size $\sigma^2_{\mathrm{WP}} \to 0^+$. This will explain the asymptotic agreement of the standard deviation of the ensemble of irrelevant weights and the target weight size in Fig. S1a, left hand side.

Summing the variances that arise at every step from Eq. (S61) yields after n steps a total additional variance

$$\langle\langle \Delta w^{\mathrm{tot}}_{\mathrm{irrel}}(n)\rangle\rangle_{\mathsf{all}} = 2\eta^2\alpha^2 \cdot (E(0) - E_{\mathrm{opt}}) \sum_{m=0}^{n-1} a^m = \frac{2\eta^2\alpha^2}{1-a}(E(0) - E_{\mathrm{opt}}) \cdot (1 - a^n). \tag{S63}$$

Here $\Delta w^{\mathrm{tot}}_{\mathrm{irrel}}(n)$ denotes the total change in the irrelevant weight up to the nth step and we used that $\langle E(m)\rangle_{\mathsf{all}} - E_{\mathrm{opt}} = (E(0) - E_{\mathrm{opt}})a^m$ for $\sigma^2_{\mathrm{WP}} \to 0^+$ (Eqs. (B38,B40)).

We can now compare the standard deviation of the irrelevant weights to the relevant weights' drift towards their targets. For this we first note that $E(0)(1 - a^n) = E(0) - \langle E(n)\rangle_{\mathsf{all}}$ and that the error at a learning step (measured against E_{opt}) is proportional to the 2-norm of the relevant weight mismatch, $E - E_{\mathrm{opt}} = \frac{1}{2}\alpha^2 \sum_{i=1}^{M} \sum_{j=1}^{N_{\mathrm{eff}}} W^2_{\mathrm{rel},ij}$, due to our assumption on the S-matrix (App. A, Sec. "Task setting"). Introducing the average of squared mismatch of the relevant weights, $\overline{W^2_{\mathrm{rel}}} = (MN_{\mathrm{eff}})^{-1} \sum_{i=1}^{M} \sum_{j=1}^{N_{\mathrm{eff}}} W^2_{\mathrm{rel},ij}$, we have $E(0) - \langle E(n)\rangle_{\mathsf{all}} = \frac{1}{2}\alpha^2 MN_{\mathrm{eff}}\left(\overline{W^2_{\mathrm{rel}}}(0) - \langle \overline{W^2_{\mathrm{rel}}}(n)\rangle_{\mathsf{all}}\right)$ and Eq. (S63) becomes

$$\langle\langle \Delta w^{\mathrm{tot}}_{\mathrm{irrel}}(n)\rangle\rangle_{\mathsf{all}} = c(\eta) \cdot \left(\overline{W^2_{\mathrm{rel}}}(0) - \langle \overline{W^2_{\mathrm{rel}}}(n)\rangle_{\mathsf{all}}\right). \tag{S64}$$

The proportionality factor $c(\eta)$ is approximately 1 at the optimal learning rate (Eq. (S57)) such that reductions in the mean square mismatch of the relevant weights co-occur with equally strong increases in the variance of each irrelevant weight; in particular the standard deviation of an irrelevant weight converges to the initial root mean squared error of the weights,

$$\sqrt{\langle\langle \Delta w^{\mathrm{tot}}_{\mathrm{irrel}}(n)\rangle\rangle_{\mathsf{all}}} \to \left(\overline{W^2_{\mathrm{rel}}}(0)\right)^{1/2} \quad \text{for } n \to \infty. \tag{S65}$$

The same then holds for the empirical standard deviation of an ensemble of irrelevant weights in a simulation.

In Fig. S1a left hand side, the learning rate is optimal, $\eta = \eta^*$; further we have $w_{\mathrm{rel},ij}(0) = 0$, $w_{\mathrm{irrel},ij}(0) = 0$ and the teacher weight strengths are all identical. Thus $\left(\overline{W^2_{\mathrm{rel}}}(0)\right)^{1/2}$ equals the teacher weight strengths. Eq. (S65) then implies that the standard deviation of the ensemble of irrelevant weights converges to the teacher weights, like the relevant weights $w_{\mathrm{rel},ij}(n)$ do. In Fig. S1a, right hand side, the learning rate is decreased by a factor of 0.1, which slows the convergence of the relevant weights down. It also decreases the proportionality factor $c(\eta)$ in Eq. (S64) (Eq. (S57)) and leads to a by a factor $\sqrt{c(\eta)} < 1$ smaller spread of irrelevant weights.

Weight diffusion for stationary error in presence of reward noise

In Fig. S1b we consider WP's weight diffusion for finite perturbation size when the final steady state error $E_{f,\mathrm{unr}} = E_f + E_{\mathrm{opt}}$ (Eqs. (S55,B38) has been reached. This minimizes the influence of the first term in Eq. (S61), since the error is minimal, and it renders the strength of the credit assignment-related update fluctuations approximately constant, since the error is approximately constant. Because the reward noise-related update fluctuations are constant as well, we have a diffusion of irrelevant weights with constant strength: The variance of the weight distribution increases in each trial by the constant specified by Eq. (S61) with $E - E_{\mathrm{opt}} = E_f$ (Eq. (S55)),

$$\begin{aligned}\langle\langle \Delta w_{\mathrm{irrel}}\rangle\rangle &\approx 2\eta^2\alpha^2 \cdot \tfrac{1}{8}\eta^2\sigma^2_{\mathrm{WP}}\alpha^6 \frac{M^3N^3_{\mathrm{eff}}}{1-a} + \tfrac{1}{4}\eta^2\sigma^2_{\mathrm{WP}}\alpha^4 \cdot M^2N^2_{\mathrm{eff}} \\ &= \left(c(\eta) + 1\right) \cdot \tfrac{1}{4}\eta^2\sigma^2_{\mathrm{eff}}\alpha^2 \cdot M^2N_{\mathrm{eff}}.\end{aligned} \tag{S66}$$

The (with respect to trial number and weight strength) homogeneous random walk or diffusion is reflected in a square root growth of the standard deviation of the sampled weight distribution in Fig. S1b, left hand side.

Networks with weight decay

In biological neural networks synaptic strengths do not diverge but stay finite. As a proof of principle, we show that multiplicative weight decay can achieve this: After each update all weights are scaled down by a factor $\gamma_{\mathrm{wd}} < 1$,

$$w_{ij}(n) = \gamma_{\mathrm{wd}} \cdot \big(w_{ij}(n-1) + \Delta w_{ij}(n-1)\big). \tag{S67}$$

Fig. S1b, right hand side, illustrates weight evolution incorporating this weight decay, after the relevant weights have reached their steady state distribution (now shifted towards smaller weights). The irrelevant weights still initially diffuse; the multiplicative weight decay, like a restoring force, counteracts and finally balances the diffusion, which leads to a steady state. We note that multiplicative weight decay is invariant with respect to rotations of the input space and thus compatible with our assumption that only the first n_{eff} inputs are nonzero (SM1, Sec. "Task relevant and irrelevant weights"). Additive weight decay, i.e. reducing (the amplitude of) each weight by the same amount, would non-isotropically bias the weight evolution.

We further note that multiplicative weight decay is on average equivalent to L2 regularization. To implement such a regularization, the error function may be modified to $E \to E + \beta \frac{1}{2} \mathrm{tr}[ww^T]$ with a parameter β specifying the regularization strength. This adds $-\beta w$ to the negative error gradient, which WP follows on average. Additive weight decay would on average be equivalent to an L1 regularization.

SM 3

Multiple subtasks

In this part we first describe the detailed setup of our tasks that are composed of multiple subtasks. Thereafter we derive the convergence factors for the learning of multiple subtasks, which were introduced in the main text using intuitive arguments, and the final error due to finite perturbations and unrealizable target components. This allows us to explain why batch learning improves the convergence speed of WP but not of NP learning. The subsequent section shows that batch learning also reduces the contribution of unrealizable target components to the final error of WP but not NP. Further we find that splitting the task into subtasks decreases the final error for learning with finite size perturbations for both WP and NP learning, while batch learning increases it.

Task setting and construction of subtasks

We adapt the framework of the analytically tractable tasks (main text, Sec. "Theoretical analysis") to tasks that consist of several subtasks. A task has input dimensionality $N_{\text{eff}}^{\text{task}}$, i.e. there are $N_{\text{eff}}^{\text{task}}$ latent inputs. The latent inputs have the same strength. In each trial a subtask of the task with dimensionality $N_{\text{eff}}^{\text{trial}}$ is presented. The error is then computed according to Eq. (8) and the updates according to Eq. (3) or Eq. (6).

Without loss of generality we assume rotated inputs. This allows to assign to each of the first $N_{\text{eff}}^{\text{task}}$ input neurons a different T-dimensional basis function e_{jt}; the $N - N_{\text{eff}}^{\text{task}}$ remaining inputs are zero. Different basis functions are orthogonal, $\frac{1}{T}\sum_{t=1}^{T} e_{jt}e_{kt} = \delta_{jk}$. In a given trial, $N_{\text{eff}}^{\text{trial}}$ of the first $N_{\text{eff}}^{\text{task}}$ inputs are chosen randomly with equal probability to be active,

$$r_{jt} = \begin{cases} \alpha \cdot e_{jt} & \text{if input } j \text{ is active} \\ 0 & \text{if input } j \text{ is inactive.} \end{cases} \tag{S68}$$

Active inputs therefore have strength α^2. For simplicity we assume that the targets are fully realizable. The inputs that are active in a subtask p therefore determine the targets via target weights w^*,

$$z_{it}^{p,*} = \sum_{j \in \Omega_p} w_{ij}^* r_{jt}^p, \tag{S69}$$

where Ω_p is the set of inputs that are active during subtask p. We refer to the inputs r_{jt}^p of subtask p as an "input pattern".

We define the error of the task as the average error over all subtasks, $E^{\text{task}} \equiv \langle E \rangle_{\text{subtasks}}$. Introducing the subtask-averaged correlation matrix $S^{\text{task}} \equiv \langle S \rangle_{\text{subtasks}}$, E^{task} can be expressed by

$$E^{\text{task}} = \langle E \rangle_{\text{subtasks}} = \langle \tfrac{1}{2} \operatorname{tr}\left[W S W^T\right] \rangle_{\text{subtasks}} = \tfrac{1}{2} \operatorname{tr}\left[W S^{\text{task}} W^T\right]. \tag{S70}$$

An input j is in a given trial active with constant probability

$$\frac{N_{\text{eff}}^{\text{trial}}}{N_{\text{eff}}^{\text{task}}} \equiv \frac{1}{P}, \tag{S71}$$

where P can be interpreted as the effective number of subtasks or patterns in the task and as the number of trials needed to gather information on all relevant weights. The first $N_{\text{eff}}^{\text{task}}$ diagonal elements of S^{task}, which are nonzero, are therefore given by $\frac{1}{P}\alpha^2$, that is for each task-relevant input by the product of its strength with its probability of being active. This stays true if the subtasks are not random but fixed such that their sets of active inputs are non-overlapping, i.e. such that each input $j \in 1, \dots, N_{\text{eff}}^{\text{task}}$ is nonzero in exactly one subtask. P is then the number of these subtasks.

The typical timescale of learning, $1/(1-a)$ (motivated after Eq. (S54)), is larger than P by a factor of $MN_{\text{eff}}^{\text{task}} + 2$ for WP and $MN_{\text{eff}}^{\text{trial}} + 2$ for NP, Eq. (15) and (16). Thus for typical task dimensions all inputs are often sampled before the error significantly changes. It is therefore not important whether random or fixed input patterns are presented and whether the presentation is in a specific order. Further, similar to the case without subtasks, it is not necessary to assume a fixed set of basis functions e_{jt} for our results to hold: The error computation and the update rules Eqs. (8,B7,B19) imply that only the weight mismatch and the correlation structure $S(n)$ of the inputs at a trial n matter for the error between student and teacher output as well as for the weight updates. In the present part we could thus also construct each trial from completely different basis functions $\tilde{e}_{jt}(n+1) \neq \tilde{e}_{jt}(n)$, as long as their correlations remain unchanged, $\frac{1}{T}\sum_{t=1}^{T} \tilde{e}_{jt}\tilde{e}_{kt} = \delta_{jk}$ (and the targets are adapted according to Eq. (S69)).

Derivation of the convergence factor

Here we derive the convergence factor of WP by considering its improvements on the current subtask and the simultaneous deterioration on all other subtasks; a slightly altered derivation holds for NP. For simplicity and without loss of generality (see the related argument at the end of the previous section) we assume that there are P fixed subtasks that are orthogonal in the sense that their sets of active inputs do not overlap. This implies that also their sets of trial-relevant weights do not overlap. As here we derive only the convergence factor, we also assume infinitesimally small perturbations and realizable targets.

The task error at trial n, Eq. (S70), is the average of the errors E_q of the subtasks q,

$$E^{\text{task}} = \frac{1}{P}\sum_{q=1}^{P} E_q(n). \tag{S72}$$

Let pattern p be presented at trial n. Then the error signal $E_p^{\text{pert}}(n) - E_p(n)$ and the update $\Delta w(n)$ depend only on pattern p, while the effect of the weight update on the task error depends on all subtasks q. After the update, the expected error for subtask p decreases by the same factor as for the single pattern case,

$$\langle E_p(n+1)\rangle = \left(1 - 2\eta\alpha^2 + \eta^2\alpha^4(MN_{\text{eff}}^{\text{trial}} + 2)\right)\cdot\langle E_p(n)\rangle \tag{S73}$$

(Eq. (B39)), while each weight that is irrelevant to the current subtask receives fluctuations with mean zero and variance

$$\langle\langle \Delta w_{\text{tr.irrel.}}(n)\rangle\rangle = \langle\langle \delta w_{\text{tr.irrel.}}^{\text{cr.as}}(n)\rangle\rangle = 2\eta^2\alpha^2\cdot E_p(n) \tag{S74}$$

(Eq. (S61) in the small σ_{NP}^2 limit). Correspondingly, the expected error for subtasks $q \neq p$ increases as

$$\begin{aligned}
\langle E_q(n+1)\rangle &= \tfrac{1}{2}\langle \text{tr}[(W + \delta w^{\text{cr.as}})S(q)(W + \delta w^{\text{cr.as}})^T]\rangle \\
&= \langle E_q(n)\rangle + \tfrac{1}{2}\langle \text{tr}[\delta w^{\text{cr.as}} S(q)(\delta w^{\text{cr.as}})^T]\rangle \\
&= \langle E_q(n)\rangle + \tfrac{1}{2}MN_{\text{eff}}^{\text{trial}}\alpha^2\cdot\langle\langle \delta w_{\text{irrel}}^{\text{cr.as}}(n)\rangle\rangle \\
&= \langle E_q(n)\rangle + MN_{\text{eff}}^{\text{trial}}\cdot\eta^2\alpha^4\cdot\langle E_p(n)\rangle.
\end{aligned} \tag{S75}$$

Inserting Eqs. (S73,S75) into Eq. (S72) yields the evolution of the task error,

$$\begin{aligned}
\langle E^{\text{task}}(n+1)\rangle &= \frac{1}{P}\cdot\left(1 - 2\eta\alpha^2 + \eta^2\alpha^4(MN_{\text{eff}}^{\text{trial}} + 2)\right)\cdot\langle E_p(n)\rangle \\
&\quad + \frac{1}{P}\sum_{q\neq p}\left(\langle E_q(n)\rangle + MN_{\text{eff}}^{\text{trial}}\cdot\eta^2\alpha^4\cdot\langle E_p(n)\rangle\right) \\
&= E^{\text{task}}(n) + \left(\frac{-2\eta\alpha^2}{P} + \frac{\eta^2\alpha^4(PMN_{\text{eff}}^{\text{trial}} + 2)}{P}\right)\cdot\langle E_p(n)\rangle.
\end{aligned} \tag{S76}$$

Assuming that $\langle E_p(n)\rangle \approx E^{\text{task}}$, i.e. that the error on all subtasks is sufficiently similar, and using $PN_{\text{eff}}^{\text{trial}} = N_{\text{eff}}^{\text{task}}$ (Eq. (S71)) finally yields

$$\langle E^{\text{task}}(n+1)\rangle = \underbrace{\left(1 - \frac{2\eta\alpha^2}{P} + \frac{\eta^2\alpha^4(MN_{\text{eff}}^{\text{task}} + 2)}{P}\right)}_{=a_{\text{WP}}}\cdot\langle E^{\text{task}}(n)\rangle. \tag{S77}$$

The factor $1/P$ arises because the changes to weights relevant in any subtask affect the error only in $1/P$ of the trials. Apart from that factor, the beneficial part of the convergence factor due to gradient following, $-\frac{1}{P}\cdot 2\eta\alpha^2$, stays the same as for a single subtask. The last part due to update fluctuations in all task-relevant weights, $\frac{1}{P}\cdot\eta^2\alpha^4(MN_{\text{eff}}^{\text{task}} + 2)$, however, increases approximately by a factor of P in relation to the beneficial part, as the number of fluctuating weights relevant for the task is P times larger than if only subtask p had to be learnt. These are the arguments given in the main text.

Minimizing a_{WP} with respect to η yields Eq. (16),

$$\eta_{\text{WP}}^* = \frac{1}{(MN_{\text{eff}}^{\text{task}} + 2)\alpha^2}, \qquad a_{\text{WP}}^* = 1 - \frac{1}{P}\frac{1}{MN_{\text{eff}}^{\text{task}} + 2}. \tag{S78}$$

For NP, the same derivation with $\langle E_q(n+1)\rangle = \langle E_q(n)\rangle$ for $q \neq p$ instead of Eq. (S75) produces Eq. (15),

$$\eta_{\text{NP}}^* = \frac{1}{(MN_{\text{eff}}^{\text{trial}} + 2)\alpha^2}, \qquad a_{\text{NP}}^* = 1 - \frac{1}{P}\frac{1}{MN_{\text{eff}}^{\text{trial}} + 2}. \tag{S79}$$

Derivation of the final error due to finite perturbations

In this section we quantify the final error in a task with multiple subtasks that results from quadratic reward noise-induced update fluctuations. The contribution of unrealizable target components will be investigated in the next section.

We consider the already described setting with a task that has effective input dimension $N_{\text{eff}}^{\text{task}}$ and is split into P non-overlapping subtasks. These have effectively $N_{\text{eff}}^{\text{trial}}$-dimensional inputs and are presented in the different learning trials. The final task error due to finite perturbation sizes may be defined as

$$E_f^{\text{task}} = \frac{1}{P} \sum_{q=1}^{P} E_{f,q}, \tag{S80}$$

where $E_{f,q}$ are the final errors of the individual subtasks, which arise due to quadratic reward noise (Eq. (S54)). We investigate how E_f^{task} depends on the number P of subtasks in which the task is split, where $P = 1$ corresponds to the single-pattern case Eqs. (S55,S56).

Combining Eqs. (S80,S54) and line (S48), the contribution of (quadratic) reward noise-induced update fluctuations to the final error is

$$E_f = \frac{1}{1-a} \cdot \frac{1}{P} \sum_{q=1}^{P} \tfrac{1}{2} \left\langle \text{tr}[\delta w^{\text{rew.noise,quad}} S[q] (\delta w^{\text{rew.noise,quad}})^T] \right\rangle . \tag{S81}$$

Because all subtasks have identical properties, we can consider the effect that an update following the presentation of an arbitrary subtask p has on the errors E_q of all subtasks q.

For WP, all weights are updated and fluctuations of the weights relevant for any subtask q (including $q = p$) are equal in size and statistics. Inserting the expressions for $\langle\langle \delta w_{ij}^{\text{rew.noise,quad}} \rangle\rangle$, η^* and a^* (Eqs. (S39,S78)) into Eq. (S81) yields

$$\begin{aligned}
E_f^{\text{WP}} &= \frac{1}{1-a^*} \cdot \frac{\alpha^2}{2P} \sum_{i=1}^{M} \sum_{j=1}^{N_{\text{eff}}^{\text{task}}} \left\langle \left(\delta w_{ij}^{\text{rew.noise,quad}} \right)^2 \right\rangle \\
&\approx P M N_{\text{eff}}^{\text{task}} \cdot \frac{\alpha^2}{2P} M N_{\text{eff}}^{\text{task}} \cdot \frac{1}{4} (\eta^*)^2 \sigma_{\text{eff}}^2 \alpha^2 M^2 N_{\text{eff}}^{\text{trial}} \\
&\approx P M N_{\text{eff}}^{\text{task}} \cdot \frac{1}{(M N_{\text{eff}}^{\text{task}} \alpha^2)^2} \cdot \frac{1}{8} \sigma_{\text{eff}}^2 \alpha^4 \frac{M^3 (N_{\text{eff}}^{\text{task}})^2}{P^2} \\
&= \frac{1}{8} \sigma_{\text{eff}}^2 M^2 N_{\text{eff}}^{\text{trial}} = \frac{1}{8} \sigma_{\text{eff}}^2 M^2 \frac{N_{\text{eff}}^{\text{task}}}{P}.
\end{aligned} \tag{S82}$$

The final error of WP thus becomes smaller if the task is split into more subtasks, even though that means that WP converges more slowly (Eq. (S78)). The reason is that WP has to operate at a roughly P times smaller learning rate, which means that update fluctuations average out over more trials, ultimately lowering the final error.

For NP, only weights relevant for the current subtask are updated such that only the term with $q = p$ in Eq. (S81) contributes. Inserting the expressions for $\langle\langle \delta w_{ij}^{\text{rew.noise,quad}} \rangle\rangle$, η^* and a^* (Eqs. (S40,S79)) into Eq. (S81) yields

$$\begin{aligned}
E_f^{\text{NP}} &= \frac{1}{1-a} \cdot \frac{\alpha^2}{2P} \sum_{i=1}^{M} \sum_{j \in \Omega_p} \left\langle \left(\delta w_{ij}^{\text{rew.noise,quad}} \right)^2 \right\rangle \\
&\approx P M N_{\text{eff}}^{\text{trial}} \cdot \frac{\alpha^2}{2P} M N_{\text{eff}}^{\text{trial}} \cdot \frac{1}{4} \eta^2 \sigma_{\text{eff}}^2 \alpha^2 M^2 T \\
&\approx M N_{\text{eff}}^{\text{task}} \cdot \frac{1}{(M N_{\text{eff}}^{\text{trial}} \alpha^2)^2} \cdot \frac{1}{8} \sigma_{\text{eff}}^2 \alpha^4 M^3 \frac{N_{\text{eff}}^{\text{task}} T}{P^2} \\
&= \frac{1}{8} \sigma_{\text{eff}}^2 M^2 T = \frac{1}{8} \sigma_{\text{eff}}^2 M^2 \frac{T^{\text{task}}}{P},
\end{aligned} \tag{S83}$$

where Ω_p is the set of inputs that are active during subtask p, T is the subtask duration and T^{task} the duration of the full task with all subtasks concatenated. We conclude that if subtasks are obtained by splitting an original, full task of duration T^{task} into P subtasks such that $T = T^{\text{task}}/P$, E_f^{NP} scales like E_f^{WP} with $1/P$. This holds despite the fact that the optimal learning rate of NP stays approximately constant when changing P. Since for any type of split $T \geq N_{\text{eff}}^{\text{trial}}$, comparison of Eq. (S83) and Eq. (S82) yields $E_f^{\text{NP}} \geq E_f^{\text{WP}}$.

Derivation of the final error due to unrealizable target components

Here we obtain the final error $E_{f,d}$ that results from the target components of a trial that are perpendicular to any of the inputs, i.e. from the *trial-unrealizable* target components. This error only occurs for NP.

We assume that in each trial the trial-unrealizable components have the same strength, leading to an unavoidable limiting error $E_{\text{opt}}^{\text{tr.unr.}}$, which is the same for each subtask. Since the error definition Eq. (S80) is normalized by P, $E_{\text{opt}}^{\text{tr.unr.}}$ is independent of P. For NP but not WP, trial-unrealizable target components add reward noise $\Delta E_{\text{pert}}^{\text{lin,unr}}$ to the error signal (Eq. (S10)), which gets translated into update fluctuations with $\langle\langle \delta w_{ij}^{\text{rew.noise,lin}} \rangle\rangle = 2\eta^2\alpha^2 E_{\text{opt}}^{\text{tr.unr.}}$ (Eq. (S37)). These results are still valid for the case considered here, with $\delta w_{ij}^{\text{rew.noise,lin}} \neq 0$ only for the $MN_{\text{eff}}^{\text{trial}}$ trial-relevant weights of the current subtask p. The update fluctuations lead to an expected increase in error (Eqs. (S47,S72)) of

$$\begin{aligned} b_{\text{NP}}^{\text{lin}} &= \frac{1}{P}\sum_{q=1}^{P} \tfrac{1}{2}\langle \text{tr}[\overbrace{\delta w^{\text{rew.noise,lin}} S_q}^{\neq 0 \text{ only for } q=p} (\delta w^{\text{rew.noise,lin}})^T]\rangle \\ &= \frac{\alpha^2}{2P}\sum_{q=1}^{P}\delta_{qp}\sum_{i=1}^{M}\sum_{j=1}^{N_{\text{eff}}^{\text{trial}}} \langle\langle \delta w_{ij}^{\text{rew.noise,lin}} \rangle\rangle \\ &= \frac{\alpha^2}{2P} MN_{\text{eff}}^{\text{trial}} \cdot 2\eta^2\alpha^2 E_{\text{opt}}^{\text{tr.unr.}} = \frac{1}{P}\eta^2\alpha^4 MN_{\text{eff}}^{\text{trial}} \cdot E_{\text{opt}}^{\text{tr.unr.}}. \end{aligned} \tag{S84}$$

The result is very similar to $b_{\text{NP}}^{\text{lin}}$ in Eq. (B44) and reproduces it for $P = 1$. When keeping $N_{\text{eff}}^{\text{task}}$ fixed and regarding P as a free parameter, both η (relative to η^*) and $N_{\text{eff}}^{\text{trial}}$ contain implicit dependencies on P. The contribution of $b_{\text{NP}}^{\text{lin}}$ to the final error (Eq. (S54)), at the optimal learning rate (Eq. (S79)), is

$$\begin{aligned} E_{f,d}(\eta^*) &\overset{\text{NP}}{=} \frac{b_{\text{NP}}^{\text{lin}}(\eta^*)}{1 - a_{\text{NP}}(\eta^*)} = \frac{1}{P}(\eta^*)^2\alpha^4 \frac{MN_{\text{eff}}^{\text{trial}}}{1 - a_{\text{NP}}(\eta^*)} \cdot E_{\text{opt}}^{\text{tr.unr.}} \\ &= \frac{1}{P}\frac{\alpha^4}{(MN_{\text{eff}}^{\text{trial}} + 2)^2\alpha^4}(MN_{\text{eff}}^{\text{trial}})P(MN_{\text{eff}}^{\text{trial}} + 2) \cdot E_{\text{opt}}^{\text{tr.unr.}} \\ &\approx E_{\text{opt}}^{\text{tr.unr.}}. \end{aligned} \tag{S85}$$

The final error contribution $E_{f,d}$ due to trial-unrealizable target components is thus (approximately) independent of the number of subtasks P that the full task is split into. Also, at the optimal learning rate, the additional error contribution for NP learning again equals the unavoidable limiting error $E_{\text{opt}}^{\text{tr.unr.}}$ (cf. Eq. (S58)).

The results of this section only hold for trial-unrealizable target components. In Sec. "Batch learning improves WP's but not NP's performance for overlapping subtasks and unrealizable target components" we make the distinction between "trial-unrealizable" and "task-unrealizable" components. There we show that trial-realizable but task-unrealizable components also harm WP learning, but that combining trials into batches renders some task-unrealizable components also trial-unrealizable, reducing the effect.

Error curves when learning multiple subtasks

Combining Eqs. (S78,S79,S82,S83,S85,S54), the errors of WP and NP evolve as

$$\langle E(n)\rangle = \left(E(0) - E_{f,\text{unr}}\right)\cdot a^n + E_{f,\text{unr}} \tag{S86}$$

to a final error $E_{f,\text{unr}} = E_f + E_{f,d} + E_{\text{opt}}^{\text{tr.unr.}}$ with

$$a_{\text{WP}}^* = 1 - \frac{1}{P}\frac{1}{MN_{\text{eff}}^{\text{task}} + 2}, \qquad a_{\text{NP}}^* = 1 - \frac{1}{P}\frac{1}{MN_{\text{eff}}^{\text{trial}} + 2}, \tag{S87}$$

$$E_f^{\text{WP}}(\eta^*) \approx \frac{1}{8}\sigma_{\text{eff}}^2 M^2 N_{\text{eff}}^{\text{trial}}, \qquad E_f^{\text{NP}}(\eta^*) \approx \frac{1}{8}\sigma_{\text{eff}}^2 M^2 T, \tag{S88}$$

$$E_{f,d}^{\text{WP}}(\eta^*) = 0, \qquad E_{f,d}^{\text{NP}}(\eta^*) \approx \frac{1}{P}\cdot E_{\text{opt}}^{\text{tr.unr.}}. \tag{S89}$$

Batch learning improves WP's but not NP's performance

Why does batch learning improve WP's but not NP's performance? In the following we obtain intuitive explanations from observing how concatenating trials into batches affects which weights are relevant and which target components are realizable in a trial.

Concatenating $K \leq P$ subtasks with non-overlapping inputs into a single batch changes the subtask parameters as follows

$$N_{\text{eff}}^{\text{trial}} \to K N_{\text{eff}}^{\text{trial}}, \qquad P \to \frac{P}{K}, \qquad \alpha^2 \to \frac{\alpha^2}{K}, \tag{S90}$$

$$N_{\text{eff}}^{\text{task}} \to N_{\text{eff}}^{\text{task}}, \qquad T \to KT. \tag{S91}$$

Here the change Eq. (S90) left reflects that the input dimensionality of K non-overlapping inputs with dimension $N_{\text{eff}}^{\text{trial}}$ is $K N_{\text{eff}}^{\text{trial}}$; consequently, the number of trial-relevant weights becomes $KM N_{\text{eff}}^{\text{trial}}$. The number of trials needed to gather information on all task-relevant weights, P, therefore decreases, by a factor of K (Eq. (S90) middle). The temporal extent of the subtask increases by a factor K (Eq. (S91) middle). The individual input vectors keep their nonzero entries and are padded by zero entries from length T to length KT. Because the entries of S have as normalizing prefactor the new total trial duration KT instead of T, the nonzero entries read α^2/K (Eq. (S90) right). Since there are K times more nonzero input vectors, the total input strength per time step, α_{tot}^2 (Eq. (A11)), remains unchanged, as we expect it from a concatenation operation. The task dimensionality is unaffected by changes of the subtasks (Eq. (S91) left).

Using these scalings and assuming $M N_{\text{eff}}^{\text{trial}} \gg 2$ in Eqs. (S78,S79) shows that, approximately, the optimal learning rate and the convergence rate ($\approx 1 - a$) of WP increase linearly with K, while the performance of NP stays unaffected,

$$\eta_{\text{WP}}^* \to \frac{1}{(M N_{\text{eff}}^{\text{task}} + 2)\frac{\alpha^2}{K}} = K \cdot \eta_{\text{WP}}^*, \qquad 1 - a_{\text{WP}}^* \to \frac{K}{P}\frac{1}{M N_{\text{eff}}^{\text{task}} + 2} = K \cdot (1 - a_{\text{WP}}^*), \tag{S92}$$

$$\eta_{\text{NP}}^* \to \frac{1}{(MK N_{\text{eff}}^{\text{trial}} + 2)\frac{\alpha^2}{K}} \approx \eta_{\text{NP}}^*, \qquad 1 - a_{\text{NP}}^* \to \frac{K}{P}\frac{1}{MK N_{\text{eff}}^{\text{trial}} + 2} \approx 1 - a_{\text{NP}}^*. \tag{S93}$$

The result can be understood as follows: In a batch of K subtasks, the error feedback of a single trial contains information on K times more weights. In WP this increases the number of weights that receive beneficial updates (Eq. (S73)) and do not simply fluctuate (Eq. (S75)), by a factor K. Therefore learning is K times faster. Since for NP trial-irrelevant weights do not fluctuate, it does not matter whether the weight update information is presented in one subtask or distributed over different ones. Therefore NP learning does not benefit from forming batches.

The scaling of the final error due to finite perturbation sizes when learning at the optimal learning rate can be obtained using Eqs. (S82,S83). If we concatenate K non-overlapping subtasks of duration T into batches, $N_{\text{eff}}^{\text{trial}} \to K N_{\text{eff}}^{\text{trial}}$ (Eq. (S90)) and $T \to KT$ (Eq. (S91)) imply

$$E_f^{\text{WP}} = \frac{1}{8}\sigma_{\text{eff}}^2 M^2 N_{\text{eff}}^{\text{trial}} \cdot K, \qquad E_f^{\text{NP}} = \frac{1}{8}\sigma_{\text{eff}}^2 M^2 T \cdot K, \tag{S94}$$

i.e. the final error increases proportional to the batch size. Fig. S5 shows WP and NP learning for different batch sizes along with predicted error curves from the results of this section. The independence of the final error in the MNIST task from the perturbation strength (Fig. S10) and for NP from the batch size (Fig. 7) suggests that final performance is not limited by the reward noise caused by finite perturbations.

Batch learning improves WP's but not NP's performance for overlapping subtasks and unrealizable targets

Are there specific effects of batch learning benefiting WP and/or NP when the subtask input patterns are overlapping and the targets contain unrealizable components? To address this questions we need to distinguish *trial-unrealizable* and *task-unrealizable* target components. We will show that trial-realizable but task-unrealizable components are a source of *gradient noise* for WP and NP, that larger batch sizes render some of these components trial-unrealizable, and that this reduces the gradient noise for WP but not NP.

Let w_{ij}^* be an optimal weight matrix that minimizes the task error Eq. (S72), such that in a subtask p and for *task-unrealizable* targets d^p the targets $z_{it}^{p,*}$ are given by

$$z_{it}^{p,*} = \sum_{j=1}^{N} w_{ij}^* r_{jt}^p + d_{it}^p, \qquad d_{it}^p = d_{it}^{p,\text{tr.real.}} + d_{it}^{p,\text{tr.unr.}}. \tag{S95}$$

Here $d^{p,\mathrm{tr.unr.}}$ is the *trial-unrealizable* component that is orthogonal to all inputs of subtask p. $d^{p,\mathrm{tr.real.}}$ is the target component that cannot be realized without simultaneously increasing the task error due to worse performance on other (overlapping) subtasks, although it could be realized in trial p. We note that $d_{it}^{p,\mathrm{tr.real.}}$ can only be nonzero if subtasks overlap. Expressing the desired output of the subtask using this distinction,

$$z_{it}^{p,*} = \sum_{j=1}^{N} w_{ij}^* r_{jt}^p + d_{it}^{p,\mathrm{tr.real.}} + d_{it}^{p,\mathrm{tr.unr.}}, \tag{S96}$$

the error for subtask p is

$$\begin{aligned} E_p &= \frac{1}{2T} \sum_{i=1}^{M} \sum_{t=1}^{T} \left(z_{it} - z_{it}^*\right)^2 = \frac{1}{2T} \sum_{i=1}^{M} \sum_{t=1}^{T} \Big(\sum_{j=1}^{N} W_{ij} r_{jt}^p - d_{it}^{p,\mathrm{tr.real.}} - d_{it}^{p,\mathrm{tr.unr.}} \Big)^2 \\ &= \tfrac{1}{2} \operatorname{tr}[W S^p W^T] - \frac{1}{T} \operatorname{tr}\left[W r^p \left(d^{p,\mathrm{tr.real.}}\right)^T\right] + \frac{1}{2T} \operatorname{tr}\left[d^p \left(d^p\right)^T\right]. \end{aligned} \tag{S97}$$

Here the weight mismatch $W = w - w^*$ is defined relative to the weights w^* that are optimal for the full task. Perturbations of the weights can couple to $d^{p,\mathrm{tr.real.}}$ but not $d^{p,\mathrm{tr.unr.}}$, while node perturbations project equally onto trial- and task-unrealizable target components. $d_{it}^{p,\mathrm{tr.real.}}$ endows the output- and weight error gradients with noise,

$$\frac{\partial E_p}{\partial z_{it}} = \underbrace{\frac{1}{T} \sum_{j=1}^{N} W_{ij} r_{jt}^p}_{\text{stoch. est. of } \frac{\partial E^{[\mathrm{task}]_{\mathrm{real}}}}{\partial z_{it}}} - \underbrace{\frac{1}{T} d_{it}^{p,\mathrm{tr.real.}}}_{\text{noise affecting WP and NP}} - \underbrace{\frac{1}{T} d_{it}^{p,\mathrm{tr.unr.}}}_{\text{noise affecting only NP}}, \tag{S98}$$

$$\frac{\partial E_p}{\partial w_{ij}} = \underbrace{\sum_{k=1}^{N} W_{ik} S_{kj}^p}_{\text{stoch. est. of } \frac{\partial E^{[\mathrm{task}]_{\mathrm{real}}}}{\partial w_{ij}}} - \underbrace{\frac{1}{T} \sum_{t=1} d_{it}^{p,\mathrm{tr.real.}} r_{jt}^p}_{\text{additional gradient noise}}. \tag{S99}$$

The first terms are stochastic estimates of the gradient of the full task with all unrealizable target components removed. The realizable part of the task is solved by the same weight configurations as the full, unrealizable task, as the unrealizable components by definition cannot be improved. Thus their contribution to the error is the same regardless of the weights, and following the gradient of the realizable task already solves the task. The other terms in Eqs. (S98,S99) therefore cause noise. $d^{p,\mathrm{tr.unr.}}$ causes reward noise and harms only NP (Eq. (S85)). $d^{p,\mathrm{tr.real.}}$ adds *gradient noise* to the weight gradient: Even if WP and NP could average over all possible perturbations and measure $\partial E_p / \partial w_{ij}$ with arbitrarily high precision, the computed weight update would not be linearly optimal. The gradient is wrong in the sense that for nonzero $d_{it}^{p,\mathrm{tr.real.}}$ the trial error gradient $\partial E_p / \partial w_{ij}$ does not provide an optimal approximation to the error gradient of the entire task. Because $d^{p,\mathrm{tr.real.}}$ is realizable within the single trial, both WP and NP try to reduce it. It therefore causes alike noise for both WP and NP.

As for tasks without subtasks (cf. SM1, Sec. "Realizable and unrealizable outputs"), the trial-unrealizable part $d^{p,\mathrm{tr.unr.}}$ affects NP by inducing reward noise. WP cannot induce output perturbation along this output component and is thus not affected. Since output perturbations directly change $E_{\mathrm{pert}} - E$ without distinguishing between realizable and unrealizable target components, $E_{\mathrm{pert}} - E$ is equally affected by $d^{p,\mathrm{tr.real.}}$ and $d^{p,\mathrm{tr.unr.}}$. Therefore basically their sum d^p matters for NP.

An increasing batch size renders some task-unrealizable but trial-realizable components also trial-unrealizable. This becomes especially apparent for the case where the batch contains all subtasks: Because there is only one (sub-)task or trial, task-unrealizable components are also trial-unrealizable, $d^{1,\mathrm{tr.unr.}} = d^1$ and $d^{1,\mathrm{tr.real.}} = 0$.

Due to the decrease of $d^{\mathrm{tr.real.}}$ for increasing batch size, the gradient noise affecting WP decreases, as it no longer induces output perturbations along these components. In contrast, the strength $\frac{1}{T} \sum_{t=1}^{T} \sum_{i=1}^{M} (d_{it}^p)^2$ of the sum $d^{p,\mathrm{tr.real.}} + d^{p,\mathrm{tr.unr.}} = d^p$, which determines NP's gradient noise, is independent of the batch size. Therefore WP but not NP benefits from increasing the batch size.

SM 4

Arbitrary input strength distributions

Error curves for input components with different strength

In the main text, Sec. “Theoretical analysis” and in App. B, Sec. “Error curves for equally strong input components” (as well as in the SM parts thereafter), we focused on the error dynamics in the special case where all latent inputs have the same strength. To understand the error dynamics in the reservoir computing simulation experiment and for completeness, here we also give the general solution for the error dynamics. We aim at expressions analogous to those of App. B, Sec. “Error curves for equally strong input components”. To obtain them we split the error E into components E^μ and replace the convergence factor a by a matrix $(A+B)_{\mu\nu}$ and the per-update error increase b by a vector b_μ.

To define the error components, we split the correlation matrix S into a sum of matrices S^μ, where each matrix contains the contribution of one eigenvalue to S,

$$S^\mu = O D^\mu O^T, \qquad D^\mu_{jk} = \alpha_\mu^2 \delta_{\mu jk}, \qquad S = \sum_{\mu=1}^{N} S^\mu, \tag{S100}$$

where O diagonalizes S and D^μ is a matrix with only one nonzero element $D^\mu_{\mu\mu} = \alpha_\mu^2$. We observe that the powers of S can be written as sums of powers of S^μ,

$$S^2 = \sum_{\mu=1}^{N} O(D^\mu)^2 O^T = \sum_{\mu=1}^{N} (S^\mu)^2, \qquad S S^\mu = S^\mu S^\mu = \alpha_\mu^2 S^\mu, \qquad \operatorname{tr}[S] = \sum_{\mu=1}^{N} \alpha_\mu^2. \tag{S101}$$

The error E then reads

$$E = \tfrac{1}{2}\operatorname{tr}[W\tilde{S}W^T] + E_{\text{opt}} = \tfrac{1}{2}\operatorname{tr}[W\sum_{\mu=1}^{N}\tilde{S}^\mu W^T] + E_{\text{opt}}. \tag{S102}$$

Here we again use the tilde symbol to distinguish the correlation matrix that stems from the cost evaluation at trial n from the correlation matrix that determines the error change due to perturbations in trial $n-1$ (App. B). Because the trace is linear, we can split the input strength-dependent part of the error into N summands E^μ as

$$E^\mu = \tfrac{1}{2}\operatorname{tr}[W\tilde{S}^\mu W^T] \qquad E = \sum_{\mu=1}^{N} E^\mu + E_{\text{opt}}. \tag{S103}$$

E^μ only depends on the strength of the μth input component and the weight mismatch projected onto the corresponding input direction (Eq. (S100)). In other words, for each input component μ we can define related weights that read out from it, and a related error component E^μ that depends on the mismatch of these weights. We now obtain a recurrence equation that relates the N error components $\langle E^\mu(n)\rangle$ at trial n to those at trial $n-1$. For this we first split $\langle E^\mu(n)\rangle$ and $\langle E^\mu(n-1)\rangle$ in Eqs. (B17, B34) up using Eq. (S103). It then suffices to also split $\tilde{S}$ up using Eq. (S100) and to explicitly evaluate the powers and traces of S using Eq. (S101). For WP we obtain

$$\begin{aligned}
\langle E^\mu(n)\rangle &\overset{\text{WP}}{=} \langle E^\mu(n-1)\rangle - \eta\operatorname{tr}[WS^\mu SW^T] + \frac{\eta^2}{2}M\operatorname{tr}[WS^2W^T]\operatorname{tr}[S^\mu] + \eta^2\operatorname{tr}[WSS^\mu SW^T] \\
&\quad + \frac{\eta^2\sigma_{\text{WP}}^2}{8}\left(M^3\operatorname{tr}[S^\mu]\operatorname{tr}[S]^2 + 2M^2\operatorname{tr}[S^\mu]\operatorname{tr}[S^2] + 4M^2\operatorname{tr}[S^\mu S]\operatorname{tr}[S] + 8M\operatorname{tr}[SS^\mu S]\right) \\
&= \langle E^\mu(n-1)\rangle - \eta\alpha_\mu^2\operatorname{tr}[WS^\mu W^T] + \frac{\eta^2}{2}M\sum_{\nu=1}^{N}\alpha_\mu^2\alpha_\nu^2\operatorname{tr}[WS^\nu W^T] + \eta^2\alpha_\mu^4\operatorname{tr}[WS^\mu W^T] \\
&\quad + \frac{\eta^2\sigma_{\text{WP}}^2}{8}\left(M^3\alpha_\mu^2\Big(\sum_{\nu=1}^{N}\alpha_\nu^2\Big)^2 + 2M^2\alpha_\mu^2\sum_{\nu=1}^{N}\alpha_\nu^4 + 4M^2\alpha_\mu^4\sum_{\nu=1}^{N}\alpha_\nu^2 + 8M\alpha_\mu^6\right) \\
&= \sum_{\nu=1}^{N}\left((1-2\eta\alpha_\mu^2+2\eta^2\alpha_\mu^4)\mathbb{1}_{\mu\nu} + \eta^2\alpha_\mu^2\alpha_\nu^2 M\right)\cdot\langle E^\nu(n-1)\rangle \\
&\quad + \tfrac{1}{8}\eta^2\sigma_{\text{WP}}^2\cdot\left(M^3\alpha_\mu^2\Big(\sum_{\nu=1}^{N}\alpha_\nu^2\Big)^2 + 2M^2\alpha_\mu^2\sum_{\nu=1}^{N}\alpha_\nu^4 + 4M^2\alpha_\mu^4\sum_{\nu=1}^{N}\alpha_\nu^2 + 8M\alpha_\mu^6\right).
\end{aligned} \tag{S104}$$

For NP we obtain

$$\begin{aligned}\langle E^\mu(n)\rangle &\overset{\text{NP}}{=} \langle E^\mu(n-1)\rangle - \eta\,\mathrm{tr}[WS^\mu SW^T] + \frac{\eta^2}{2}M\,\mathrm{tr}[WSW^T]\,\mathrm{tr}[S^\mu S] + \eta^2\,\mathrm{tr}[WSS^\mu SW^T] \\ &\quad + \frac{\eta^2\sigma_{\text{NP}}^2}{8T}\,\mathrm{tr}[S^\mu S]\cdot\left(M^3T^2+6M^2T+8M\right) + \eta^2 M\,\mathrm{tr}[S^\mu S]\cdot\frac{1}{2T}\,\mathrm{tr}[dd^T] \\ &= \langle E^\mu(n-1)\rangle - \eta\alpha_\mu^2\,\mathrm{tr}[WS^\mu W^T] + \frac{\eta^2}{2}\alpha_\mu^4 M\sum_{\nu=1}^{N}\mathrm{tr}[WS^\nu W^T] + \eta^2\alpha_\mu^4\,\mathrm{tr}[WS^\mu W^T] \\ &\quad + \frac{\eta^2\sigma_{\text{NP}}^2}{8T}\alpha_\mu^4\cdot\left(M^3T^2+6M^2T+8M\right) + \eta^2 M\alpha_\mu^2\,\mathrm{tr}[S^\mu]\cdot E_{\text{opt}} \\ &= \sum_{\nu=1}^{N}\left((1-2\eta\alpha_\mu^2+2\eta^2\alpha_\mu^4)\mathbb{1}_{\mu\nu}+\eta^2\alpha_\mu^4 M\right)\cdot\langle E^\nu(n-1)\rangle \\ &\quad + \tfrac{1}{8}\eta^2\sigma_{\text{NP}}^2\alpha_\mu^4\cdot\left(M^3T+6M^2+8\frac{M}{T}\right) + \eta^2\alpha_\mu^4 M\cdot E_{\text{opt}}.\end{aligned} \tag{S105}$$

Both relations can be written as

$$\langle E^\mu(n)\rangle = \sum_{\nu=1}^{N}(A+B)_{\mu\nu}\cdot\langle E^\nu(n-1)\rangle + b_\mu, \tag{S106}$$

where A is the same for WP and NP but B and b differ,

$$A_{\mu\nu} = (1-2\eta\alpha_\mu^2+\eta^2\alpha_\mu^4)\mathbb{1}_{\mu\nu}, \tag{S107}$$

$$B_{\mu\nu}^{\text{WP}} = \eta^2\alpha_\mu^2\alpha_\nu^2 M + \eta^2\alpha_\mu^4\delta_{\mu\nu}, \tag{S108}$$

$$B_{\mu\nu}^{\text{NP}} = \eta^2\alpha_\mu^4 M + \eta^2\alpha_\mu^4\delta_{\mu\nu}, \tag{S109}$$

$$b_\mu^{\text{WP}} = \tfrac{1}{8}\eta^2\sigma_{\text{WP}}^2\cdot\left(M^3\alpha_\mu^2\Big(\sum_{\nu=1}^{N}\alpha_\nu^2\Big)^2 + 2M^2\alpha_\mu^2\sum_{\nu=1}^{N}\alpha_\nu^4 + 4M^2\alpha_\mu^4\sum_{\nu=1}^{N}\alpha_\nu^2 + 8M\alpha_\mu^6\right), \tag{S110}$$

$$b_\mu^{\text{NP}} = \tfrac{1}{8}\eta^2\sigma_{\text{NP}}^2\alpha_\mu^4\cdot\left(M^3T+6M^2+8\frac{M}{T}\right) + \eta^2\alpha_\mu^4 M\cdot E_{\text{opt}}. \tag{S111}$$

The matrix A is diagonal, but B is not and mixes the different error components. A originates from the effect of the mean update (cf. Eqs. (S19,S45)), and B from that of credit-assignment-related update fluctuations (Eqs. (S20,S46)). $B_{\mu\nu}$ measures the increase in error component E^μ due to update fluctuations $\delta w^{\text{cr.as}}$ caused by output perturbations parallel to the νth input component. The diagonal entries $\eta^2\alpha_\mu^4\delta_{\mu\nu}$ in Eqs. (S108,S109) arise due to correlations between the corresponding update and error signal components. The different form of $B_{\mu\nu}$ for WP and NP can be understood by considering three factors: First, in WP the strength (variance) of the induced output perturbation along the νth input component is proportional to the strength α_ν^2 of the νth input component, while for NP output perturbations have the same size along all directions. Second, in NP the update of the weights that read out from the μth input component are constructed by projecting the output perturbations onto that input component in the eligibility trace (Eq. (6)). The update of weights that read out from the μth input component thus scales with its strength α_μ^2. WP, on the other hand, constructs updates by multiplying weight perturbations with the same error signal for all weights. Third, the effect of update noise on error component E_μ scales with the related input strength α_μ^2 for both WP and NP. Together, $B_{\mu\nu}$ thus scales with $\alpha_\mu^2\alpha_\nu^2$ for WP and with α_μ^4 for NP.

Eq. (S106) is solved by

$$\begin{aligned}\langle E^\mu(n)\rangle &= \sum_{\nu=1}^{N}\left(A+B\right)_{\mu\nu}^{n}\langle E^\nu(0)\rangle + \sum_{\nu=1}^{N}\sum_{s=0}^{n-1}\left(A+B\right)_{\mu\nu}^{s} b_\nu \\ &= \sum_{\nu=1}^{N}\left(A+B\right)_{\mu\nu}^{n}\langle E^\nu(0)\rangle + \sum_{\nu=1}^{N}\left(\left(\mathbb{1}-\left(A+B\right)\right)^{-1}\left(\mathbb{1}-\left(A+B\right)^{n}\right)\right)_{\mu\nu} b_\nu.\end{aligned} \tag{S112}$$

For WP, $A+B$ is symmetric (Eq. (S108)). We can therefore diagonalize it and express the error evolution in terms of error modes that decay independently of each other. This is not possible for NP where $A+B$ is in general non-normal (Eq. (S109)). Therefore, even for the considered convex error function, the error of NP can initially increase due to transient non-normal amplification. The different properties of $A+B$ suggest to perform a diagonalization of $A+B$ for WP and a Schur decomposition for NP to analyze the error evolution of specific networks. We will, however, continue to work in the space of error components and not error modes (the eigen- or Schur vectors of $A+B$) because of the simpler expressions and mechanistic interpretations.

For a fair comparison we set $\sigma^2_{\mathrm{NP}} = \sigma^2_{\mathrm{eff}}$ and $\sigma^2_{\mathrm{WP}} = \sigma^2_{\mathrm{eff}} / \sum_{\nu=1}^N \alpha_\nu^2$ (App. A, Sec. "Effective perturbation strength") and express the recursion in terms of the effective perturbation strength. This affects only b,

$$b_\mu^{\mathrm{WP}} = \tfrac{1}{8}\eta^2\sigma^2_{\mathrm{eff}} \cdot \Big(M^3\alpha_\mu^2 \sum_{\nu=1}^N \alpha_\nu^2 + 2M^2\alpha_\mu^2 \frac{\sum_{\nu=1}^N \alpha_\nu^4}{\sum_{\nu=1}^N \alpha_\nu^2} + 4M^2\alpha_\mu^4 + 8M \frac{\alpha_\mu^6}{\sum_{\nu=1}^N \alpha_\nu^2} \Big), \tag{S113}$$

$$b_\mu^{\mathrm{NP}} = \tfrac{1}{8}\eta^2\sigma^2_{\mathrm{eff}}\alpha_\mu^4 \cdot \Big(M^3 T + 6M^2 + 8\frac{M}{T} \Big) + \eta^2\alpha_\mu^4 M \cdot E_{\mathrm{opt}}. \tag{S114}$$

Evolution of error components related to strong and weak inputs

For infinitesimal perturbation strength and realizable targets, the amount of interference between error components determines the differences in the convergence behavior of the learning rules (cf. the identical diagonal elements of $A + B$ Eq. (S107-S109)). Whether WP or NP generates more interference to an error component E_μ depends on the concrete distribution of error components in the considered learning step. This distribution, in turn, depends on the initial conditions of the training.

In order to analyze the effect of interference on the convergence of error components related to strong and weak input strengths, we consider a single update. We assume that σ_{eff} is negligible and that the target is realizable, $d = 0$ and $E_{\mathrm{opt}} = 0$. This implies $b = 0$ (Eqs. (S113,S114)) and the error decay is determined by $A + B$ (Eq. (S106)). Inserting the expressions for $B^{\mathrm{WP|NP}}$, Eqs. (S108,S109), into Eq. (S106) yields

$$\langle E^\mu(n)\rangle \overset{\mathrm{WP}}{=} (A_{\mu\mu} + \eta^2\alpha_\mu^4) \cdot \langle E^\mu(n-1)\rangle + \eta^2\alpha_\mu^2 M \cdot \sum_{\nu=1}^N \alpha_\nu^2 \langle E^\nu(n-1)\rangle \tag{S115}$$

$$\langle E^\mu(n)\rangle \overset{\mathrm{NP}}{=} (A_{\mu\mu} + \eta^2\alpha_\mu^4) \cdot \langle E^\mu(n-1)\rangle + \underbrace{\eta^2\alpha_\mu^2 M \cdot \alpha_\mu^2 \sum_{\nu=1}^N \langle E^\nu(n-1)\rangle}_{\equiv \Delta E^{\mu,\mathrm{interference}}_{\mathrm{upd}}}. \tag{S116}$$

The last terms describe the error increase of component E^μ due to interference with all other components E^ν. Here we compare how the error components of WP and NP evolve after a single update when starting from the same distribution. Consider the ratio of the above interference terms,

$$\frac{\Delta E^{\mu,\mathrm{interference}}_{\mathrm{NP,upd}}}{\Delta E^{\mu,\mathrm{interference}}_{\mathrm{WP,upd}}} = \frac{\alpha_\mu^2 \sum_{\nu=1}^N \langle E^\nu(n-1)\rangle}{\sum_{\nu=1}^N \alpha_\nu^2 \langle E^\nu(n-1)\rangle} = \frac{\alpha_\mu^2}{\alpha_c^2} \quad \begin{cases} > 1: & E^\mu \text{ receives less interference for WP,} \\ < 1: & E^\mu \text{ receives less interference for NP,} \end{cases} \tag{S117}$$

where we defined the critical input strength

$$\alpha_c^2 = \frac{\sum_{\nu=1}^N \alpha_\nu^2 \langle E^\nu(n-1)\rangle}{\sum_{\nu=1}^N \langle E^\nu(n-1)\rangle}. \tag{S118}$$

Eq. (S117) implies that components connected to strong inputs with $\alpha_\mu^2 > \alpha_c^2$ are learned faster for WP while components related to weak inputs with $\alpha_\mu^2 < \alpha_c^2$ improve faster for NP. In particular, the largest input component always improves faster or equally fast for WP compared to NP, while the reverse holds for the the smallest input component (in the absence of reward noise).

In networks that are highly noisy (like biological ones) the task output needs to be generated by sufficiently strong latent input components. In other words, there needs to be an input representation that fits the task and clearly exceeds the noise. For initially homogeneously distributed weights this means that the weights related to the strongest input components typically produce the largest errors. Their faster convergence for WP indicates that WP may typically improve faster than NP, at least in the beginning of learning. Towards the end of learning, for fine-tuning, also modifications of weights reading from weaker inputs may be important such that NP becomes faster (Fig. S7).

Final weight spread

By projecting the weight matrix onto an input component $\tilde{r}_{\mu t}$ (SM1, Sec. "Task relevant and irrelevant weights"), we can define related weights W^μ that read out from it and whose mismatch determines the corresponding error component E^μ. Eq. (S103) then becomes

$$E^\mu = \tfrac{1}{2}\,\mathrm{tr}[W\tilde{S}^\mu W^T] = \tfrac{1}{2}\,\mathrm{tr}[W^\mu\tilde{S}^\mu(W^\mu)^T] = \tfrac{1}{2}\alpha_\mu^2|W^\mu|^2. \tag{S119}$$

The squared norm of the weight mismatch, $|W^\mu|^2$, is thus proportional to the related error component (Eq. (S106)) divided by the input strength α_μ^2. If the expectation value W_{ij}^μ is zero, as in all experiments in main text, Sec. "Theoretical analysis", except when there is weight decay or input noise (which bias w_{ij}^μ towards zero), then $\langle |W^\mu|^2 \rangle$ is the variance of the weights w^μ.

We now estimate from Eqs. (S106–S111) how the final squared weight mismatch depends on the related input strength. When assuming large M and T and neglecting the self-interaction terms in A and B, we can approximate A, B and b by their leading order terms

$$A_{\mu\nu} \approx (1 - 2\eta\alpha_\mu^2 + \cancel{\eta^2\alpha_\mu^4})\,\mathbb{1}_{\mu\nu}, \tag{S120}$$

$$B_{\mu\nu}^{\mathrm{WP}} \approx \eta^2\alpha_\mu^2\alpha_\nu^2 M + \cancel{\eta^2\alpha_\mu^4\delta_{\mu\nu}}, \tag{S121}$$

$$B_{\mu\nu}^{\mathrm{NP}} \approx \eta^2\alpha_\mu^4 M + \cancel{\eta^2\alpha_\mu^4\delta_{\mu\nu}}, \tag{S122}$$

$$b_\mu^{\mathrm{WP}} \approx \tfrac{1}{8}\eta^2\sigma_{\mathrm{WP}}^2 \cdot \Big(M^3\alpha_\mu^2\Big(\sum_{\nu=1}^N \alpha_\nu^2\Big)^2 + \cancel{2M^2\alpha_\mu^2\sum_{\nu=1}^N \alpha_\nu^4 + 4M^2\alpha_\mu^4\sum_{\nu=1}^N \alpha_\nu^2 + 8M\alpha_\mu^6}\Big), \tag{S123}$$

$$b_\mu^{\mathrm{NP}} \approx \tfrac{1}{8}\eta^2\sigma_{\mathrm{NP}}^2\alpha_\mu^4 \cdot \Big(M^3T + \cancel{6M^2 + 8\frac{M}{T}}\Big) + \eta^2\alpha_\mu^4 M \cdot E_{\mathrm{opt}}. \tag{S124}$$

With these approximations, the expected evolution of the error components (Eq. (S106)) shows a simple dependence on the input strength α_μ^2, as the dependence on ν factors out. For WP we find

$$\begin{aligned}\langle E^\mu(n)\rangle &\overset{\mathrm{WP}}{\approx} (1 - 2\eta\alpha_\mu^2)\,\langle E^\mu(n-1)\rangle + \eta^2\alpha_\mu^2 M \sum_{\nu=1}^N \alpha_\nu^2\,\langle E^\nu(n-1)\rangle + (\tfrac{1}{8}\eta^2\sigma_{\mathrm{WP}}^2 M^3\alpha_\mu^2)\cdot\Big(\sum_{\nu=1}^N \alpha_\nu^2\Big)^2 \\ &= \langle E^\mu(n-1)\rangle + \alpha_\mu^2 \cdot \underbrace{\Bigg(-2\eta\langle E^\mu(n-1)\rangle + \overbrace{\eta^2 M\sum_{\nu=1}^N \alpha_\nu^2\,\langle E^\nu(n-1)\rangle + \tfrac{1}{8}\eta^2\sigma_{\mathrm{WP}}^2 M^3\cdot\Big(\sum_{\nu=1}^N \alpha_\nu^2\Big)^2}^{\text{independent of }\mu}\Bigg)}_{=\,0\text{ after convergence if }\alpha_\mu\neq 0}.\end{aligned} \tag{S125}$$

After convergence $\langle E^\mu(n)\rangle = \langle E^\mu(n-1)\rangle$ such that either $\alpha_\mu = 0$ or the highlighted term inside the brackets must be zero. For relevant weights we can thus equate the bracket to zero and solve for the first $\langle E^\mu(n-1)\rangle$ inside it. This reveals that $\langle E^\mu(n\to\infty)\rangle$ is independent of μ and together with Eq. (S119) the scaling of $\langle |W^\mu(n\to\infty)|^2\rangle$ with α_μ,

$$\langle E^\mu(n\to\infty)\rangle \text{ is independent of } \alpha_\mu^2, \qquad \langle |W^\mu(n\to\infty)|^2\rangle \propto \frac{1}{\alpha_\mu^2}. \qquad \text{(WP)} \tag{S126}$$

We conclude that for WP, each error component E^μ adds the same contribution to the final error, regardless of the strength α_μ^2 of its corresponding input component, unless it is exactly zero. Eq. (S126) further states that for nonzero input components the squared norm $\langle |W^\mu|^2\rangle$ after convergence, or the weights' variance, scales inversely with the input strengths α_μ^2. In particular, weights associated with weak inputs will diffuse strongly but settle with a large, finite variance. Only input components that are exactly zero contribute $E^\mu = 0$ to the error. Their weights are completely irrelevant and diffuse to an infinitely broad distribution. In the main text, Sec. "Input noise", and in the next Sec. "Small input components" we argue that small but nonzero input components can still be practically irrelevant if the length of the training and/or the weight strength are limited.

For NP, Eq. (S106) simplifies to

$$\begin{aligned}\langle E^\mu(n)\rangle &\overset{\mathrm{NP}}{\approx} (1 - 2\eta\alpha_\mu^2)\,\langle E^\mu(n-1)\rangle + \eta^2\alpha_\mu^4 M\langle E(n-1)\rangle + \tfrac{1}{8}\eta^2\sigma_{\mathrm{NP}}^2\alpha_\mu^4 M^3 T + \eta^2\alpha_\mu^4 M\cdot E_{\mathrm{opt}} \\ &= \langle E^\mu(n-1)\rangle \underbrace{-\alpha_\mu^2\cdot 2\eta\langle E^\mu(n-1)\rangle + \alpha_\mu^4\cdot\overbrace{\Big(\eta^2 M\langle E(n-1)\rangle + \tfrac{1}{8}\eta^2\sigma_{\mathrm{NP}}^2 M^3 T + \eta^2 M\cdot E_{\mathrm{opt}}\Big)}^{\text{independent of }\mu}}_{=\,0\text{ after convergence}}\end{aligned} \tag{S127}$$

Again the highlighted terms cancel after convergence of the error and can for nonzero input strength α_μ^2 be solved for $\langle E^\mu(n\to\infty)$. This yields

$$\langle E^\mu(n\to\infty)\rangle \propto \alpha_\mu^2, \qquad \langle |W^\mu(n\to\infty)|^2\rangle \text{ is independent of } \alpha_\mu^2. \qquad \text{(NP)} \tag{S128}$$

We conclude that, in contrast to WP, error components related to weak input components contribute little to the final error, while those related to strong input components contribute most. Eq. (S128) further shows that for NP the final variance of the weight distribution is independent of α_μ^2; each weight is learned with the same precision.

Small input components

Strictly speaking, weights are completely irrelevant only if the inputs that they read out from are exactly zero (assuming without loss of generality rotated inputs). This is because changing the weights of any nonzero input has some effect on the output. Similarly, target components are completely unrealizable only if they are not present at all in the input. This raises the question whether our findings generalize to small but nonzero inputs.

To address it, we note that weights that read out from small inputs need to be large to have a sizeable effect on output and error. Indeed, Eq. (S119) shows that the size of the weight mismatch necessary to cause an error contribution E^μ is inversely proportional to the input strength, $|W^\mu|^2 \sim \alpha_\mu^{-2}$. For a given error tolerance, this means that the contribution from weights that read out weak inputs can be neglected as long as these *effectively irrelevant* weights remain small enough.

One setting in which these weights remain small arises if the training duration is long enough for the relevant weights to converge, but too short for the effectively irrelevant weights to noticeably contribute to the error. Fig. 6c shows such a scenario, in which the weak inputs are given by white input noise.

Weight decay can also contain the growth of effectively irrelevant weights (Fig. 2bii,S1b, SM2, Sec. “Networks with weight decay”). We expect this to work particularly well if there is a separation of timescales: if the convergence time of relevant weights is shorter than the time after which the effectively irrelevant weights make a sizeable contribution to the error, then the weight decay can operate on the longer timescale and only weakly affect the relevant weights (Fig. 6c). Introducing an upper bound for the magnitude of individual weights can similarly limit the error contribution from effectively irrelevant weights. Depending on the task setting, there may be a bound that is both large enough to not limit the relevant weights and small enough so that effectively irrelevant weights can be neglected.

SM 5

Input and perturbation correlations

Invariance of GD, WP and NP to temporal correlations and task reordering

The learning of GD, WP and NP is not affected by correlations in the inputs. More precisely: the probability distribution of learning curves does not depend on the temporal correlations of the stochastic process that the input vectors and the additional, unrealizable target components are drawn from (Fig. S3), but only on its one-dimensional finite-dimensional distributions. GD and WP's independence of temporal input and target correlations as well as of reordering or permuting the task originates from the fact that they depend only on the current weight mismatch W, the matrix S of instantaneous input correlations and (WP only) on the applied noise ξ^{WP} (Eqs. (Eqs. (A14,B1,B6,B7))). W and ξ^{WP} do not depend on the input and S_{jk} is not affected by relations between r_{jt} and r_{ks} with $s \neq t$. NP additionally depends on the projections of ξ^{NP} on the input r and the unrealizable target component d (Eqs. (A14,B18,B19)). The probability distributions of these projections are not affected by temporal correlations in r_{jt} and d_{it}, because the iid entries of the ith perturbation vector ξ^{NP}_{it} jointly have a T-dimensional, rotationally symmetric Gaussian distribution. Its projection on a constant vector is thus independent of the direction of the vector. In more detail:

- The error of GD, Eq. (8), depends on W and S. As as consequence, the GD weight update depends on W and S.
- The error of WP, Eq. (8), depends on W and S. The error of WP after the perturbation, Eq. (B6), depends on W, S and ξ^{WP}. As a consequence, the WP weight update, Eq. (B7) depends on W, S and ξ^{WP}.
- The error of NP, Eq. (8), depends on W and S. The error of NP after the perturbation, Eq. (B18), depends on W and the projections $\sum_t r_{jt}\xi^{\mathrm{NP}}_{jt}$ and $\sum_t d_{jt}\xi^{\mathrm{NP}}_{jt}$. The eligibility trace is $\sum_t r_{jt}\xi^{\mathrm{NP}}_{jt}$. The NP weight update Eq. (B19) therefore depends on W, ξ^{NP} and the projections $\sum_t r_{jt}\xi^{\mathrm{NP}}_{jt}$ and $\sum_t d_{jt}\xi^{\mathrm{NP}}_{jt}$.

Reordering and correlations within the input activity at different times do not affect S (and W and ξ^{WP}). Therefore they do not affect the learning process of GD and WP. Further, reordering and correlations do not affect the probability distributions of the projections of ξ^{NP} on r and d. The former holds because, the ξ^{NP}_{it} are are at different time points identically distributed. Temporal correlations in r_{jt} and d_{it} leave the distributions invariant, because the iid entries of the ith perturbation vector ξ^{NP}_{it} jointly have a T-dimensional, rotationally symmetric Gaussian distribution. Its projection on a constant vector is thus independent of the direction of the vector. Fig. S3 illustrates the invariance of the learning curves with respect to the introduction of input correlations by numerical simulations.

NPc: Learning with temporally correlated node perturbations

This section introduces and discusses in more detail NPc, which learns time-correlated tasks with time-correlated node perturbations. To obtain the correlated node perturbations we draw the initial perturbation at $t = 0$ and create later perturbations according to

$$\xi^{\mathrm{NPc}}_{i0} = \sigma_{\mathrm{eff}} \cdot \tilde{\xi}_{i0}, \qquad \xi^{\mathrm{NPc}}_{it+1} = \gamma \xi^{\mathrm{NPc}}_{it} + \sqrt{1-\gamma^2}\sigma_{\mathrm{eff}} \cdot \tilde{\xi}_{it+1}. \tag{S129}$$

Here $\tilde{\xi}_{it}$ is Gaussian white noise,

$$\tilde{\xi}_{it} \sim \mathcal{N}(0,1), \qquad \langle \xi^{\mathrm{NPc}}_{it} \xi^{\mathrm{NPc}}_{mt+s} \rangle = \delta_{im}\gamma^{|s|}\sigma^2_{\mathrm{eff}}, \tag{S130}$$

and the factor $\gamma = \exp(-1/\tau_{\mathrm{NPc}})$ determines the correlation time τ_{NPc}. This can be linked to an effective time dimension T_{eff},

$$T^{\mathrm{pert}}_{\mathrm{eff}} = \frac{T}{\tau_{\mathrm{NPc}}+1}, \qquad \gamma = \exp\left(-\frac{T^{\mathrm{pert}}_{\mathrm{eff}}}{T - T^{\mathrm{pert}}_{\mathrm{eff}}}\right). \tag{S131}$$

For $T^{\mathrm{pert}}_{\mathrm{eff}} = T$, we define $\gamma = \tau_{\mathrm{NPc}} = 0$; in this case NPc is exactly identical to NP. A low effective (temporal) perturbation dimension $T^{\mathrm{pert}}_{\mathrm{eff}}$ means that the perturbations' variance concentrates on only few components.

SM 6

Improved learning rules

The new insights into the mechanisms of WP and NP gained through this work enable the construction of related, under certain conditions more powerful learning rules. We note that also NPc (cf. main text, section "Input and perturbation correlations" and SM5) may be considered as such an improved method. In the current part we describe in detail the modified WP rule WP0 and hybrid perturbation (HP). WP0 is useful if the input is sparse, HP is useful if the inputs have the same strength.

WP0: Assign zero credit to zero inputs

The WP scheme has no direct way of solving the credit assignment problem of finding the weight perturbations that were responsible for causing the error signal $\Delta E^{\text{lin}}_{\text{pert}}$. Therefore it updates all weights, with the consequence that irrelevant weights diffuse and convergence slows down when multiple input patterns have to be learned.

There is a straightforward way to improve credit assignment for the case that some inputs are zero (or negligible), because then the corresponding weights do not affect the output and error - they cannot be credited for any reward changes. This information, i.e. the absence of presynaptic input, is locally available to synapses and its use means incorporation of model information, promising to improve the learning rule.

Using these observations, the update equation of the improved learning rule WP0 reads

$$\Delta w_{ij}^{\text{WP0}} = \begin{cases} 0 & \text{if } r_{jt} = 0 \ \forall t, \\ -\frac{\eta}{\sigma^2_{\text{WP}}}\left(E^{\text{pert}} - E\right)\xi_{ij}^{\text{WP}} & \text{else,} \end{cases} \tag{S132}$$

where the weight perturbations of WP0 are drawn from the same distribution as the ξ_{ij}^{WP} of WP. We note that output nonlinearities with plateaus, i.e. $g'(y) = 0$ for total weighted inputs y in some range, allow to further improve WP0 in alike manner: the weight w_{ij} should not be changed if $g'(y_{it})r_{jt} = 0 \ \forall t$, since the tried small perturbation ξ_{ij}^{WP} of w_{ij} cannot have influenced the output and the error. Neurons with such nonlinearities are for example rate neurons with ReLU activation functions or spiking neurons with a spike threshold.

In practice, inputs and g' may not be exactly zero. One can then introduce a threshold below which the weights are not updated. The largest improvements of WP0 over WP are expected if coding is sparse (many inputs are zero) and only a subset of postsynaptic neurons is non-saturated (has $g' \neq 0$).

Hybrid perturbation (HP): Using WP to produce output perturbations and NP to produce updates

WP and NP have different advantages: the output perturbations of WP lie completely in the realizable subspace and do not interfere with unrealizable target components, while NP's use of eligibility traces lets it solve part of the credit assignment problem such that irrelevant weights do not diffuse and NP converges faster than WP when multiple input patterns have to be learned. We here aim to combine the advantageous features of both learning rules into one rule, HP.

To this end, output perturbations are induced by perturbing the weights like in WP,

$$z_{it}^{\text{pert,HP}} = \sum_{j=1}^{N}(w_{ij} + \xi_{ij}^{\text{WP}})r_{jt}, \tag{S133}$$

where $g(\cdot) = \text{Id}(\cdot)$ and we named the weight perturbations of HP ξ_{ij}^{WP}, as they are drawn from the same distribution as for WP. Thus they do not interfere with unrealizable target components, which only shift the final error of HP by E_{opt} as for WP (Eq. (S54) and Fig. 4). They induce output perturbations of the form $\xi_{it}^{\text{out}} = \sum_{j=1}^{N}\xi_{ij}^{\text{WP}}r_{jt}$, which are used to calculate NP-like updates by using eligibility traces,

$$\begin{aligned} \Delta w_{ij}^{\text{HP}} &= -\frac{\eta}{\sigma^2_{\text{WP}}T}(E^{\text{pert}} - E)\sum_{t=1}^{T}\xi_{it}^{\text{out}}r_{jt} \\ &= -\frac{\eta}{\sigma^2_{\text{WP}}T}(E^{\text{pert}} - E)\sum_{t=1}^{T}\sum_{k=1}^{N}\xi_{ik}^{\text{WP}}r_{kt}r_{jt} \\ &= -\frac{\eta}{\sigma^2_{\text{WP}}}(E^{\text{pert}} - E)\sum_{k=1}^{N}\xi_{ik}^{\text{WP}}S_{kj}. \end{aligned} \tag{S134}$$

The normalization $1/(\sigma_{\mathrm{WP}}^2 T)$ contains an additional factor $1/T$. The perturbed error is the same as for WP,

$$E^{\mathrm{pert}} = \tfrac{1}{2}\,\mathrm{tr}[(W+\xi^{\mathrm{WP}})S(W+\xi^{\mathrm{WP}})^T] + E_{\mathrm{opt}} = E + \mathrm{tr}[WS\xi^{\mathrm{WP}^T}] + \tfrac{1}{2}\,\mathrm{tr}[\xi^{\mathrm{WP}}S\xi^{\mathrm{WP}^T}], \tag{S135}$$

such that the update reads

$$\Delta w_{ij} = -\frac{\eta}{\sigma_{\mathrm{WP}}^2}\left(\,\mathrm{tr}[WS\xi^{\mathrm{WP}^T}] + \tfrac{1}{2}\,\mathrm{tr}[\xi^{\mathrm{WP}}S\xi^{\mathrm{WP}^T}]\right)\sum_{k=1}^{N}\xi_{ik}^{\mathrm{WP}}S_{kj}. \tag{S136}$$

For brevity, we now only consider the linear components of the error signal (i.e. $\mathrm{tr}[WS\xi^{\mathrm{WP}^T}]$) that determine the convergence speed and behavior, neglecting the reward noise $\Delta E_{\mathrm{pert}}^{\mathrm{quad}} = \frac{1}{2}\,\mathrm{tr}[\xi^{\mathrm{WP}}S\xi^{\mathrm{WP}^T}]$. (This is equivalent to learning in the small σ_{eff} limit; realizability of targets does not affect HP anyways.) Then the μth component of the expected error after an update is

$$\langle E^\mu(n+1)\rangle = \langle E^\mu(n)\rangle + \langle \mathrm{tr}[WS^\mu\Delta w^T]\rangle + \tfrac{1}{2}\langle \mathrm{tr}[\Delta w S^\mu \Delta w^T]\rangle \tag{S137}$$

(Eq. (S103)), where the effect of the mean update is

$$\begin{aligned}\langle \mathrm{tr}[WS^\mu\Delta w^T]\rangle &= -\frac{\eta}{\sigma_{\mathrm{WP}}^2}\langle \mathrm{tr}[WS^\mu S\xi^{\mathrm{WP}^T}]\,\mathrm{tr}[WS\xi^{\mathrm{WP}^T}]\rangle \\ &= -\frac{\eta}{\sigma_{\mathrm{WP}}^2}\sum_{im=1}^{M}\sum_{jklpq=1}^{N} W_{ij}S_{jk}^\mu S_{kl}W_{mp}S_{pq}\langle \xi_{il}^{\mathrm{WP}}\xi_{mq}^{\mathrm{WP}}\rangle \\ &= -\eta\,\mathrm{tr}[WS^\mu S^2 W^T] = -2\eta\alpha_\mu^4\cdot\langle E^\mu(n)\rangle.\end{aligned} \tag{S138}$$

The quadratic contributions, which describe update fluctuations due to the credit assignment problem and which are responsible for slowing down the learning, are

$$\begin{aligned}\tfrac{1}{2}\langle \mathrm{tr}[\Delta w S^\mu \Delta w^T]\rangle &= \frac{\eta^2}{\sigma_{\mathrm{WP}}^4}\tfrac{1}{2}\langle \mathrm{tr}[WS\xi^{\mathrm{WP}^T}]\,\mathrm{tr}[\xi^{\mathrm{WP}}SS^\mu S\xi^{\mathrm{WP}^T}]\,\mathrm{tr}[WS\xi^{\mathrm{WP}^T}]\rangle \\ &= \frac{\eta^2}{2\sigma_{\mathrm{WP}}^4}\sum_{imn=1}^{M}\sum_{jkpqrstu=1}^{N}\langle W_{ij}S_{jk}\xi_{ik}^{\mathrm{WP}}\xi_{mp}^{\mathrm{WP}}S_{pq}S_{qr}^\mu S_{rs}\xi_{ms}^{\mathrm{WP}}W_{nt}S_{tu}\xi_{nu}^{\mathrm{WP}}\rangle \\ &= \frac{\eta^2}{2\sigma_{\mathrm{WP}}^4}\sum_{imn=1}^{M}\sum_{jkpqrstu=1}^{N} W_{ij}S_{jk}S_{pq}S_{qr}^\mu S_{rs}W_{nt}S_{tu}\langle \xi_{ik}^{\mathrm{WP}}\xi_{mp}^{\mathrm{WP}}\xi_{ms}^{\mathrm{WP}}\xi_{nu}^{\mathrm{WP}}\rangle.\end{aligned} \tag{S139}$$

Using $\langle \xi_{ij}^{\mathrm{WP}}\xi_{mk}^{\mathrm{WP}}\rangle = \sigma_{\mathrm{WP}}^2\delta_{im}\delta_{jk}$ and Isserli's theorem, the appearing moment of ξ^{WP} evaluates to

$$\langle \xi_{ik}^{\mathrm{WP}}\xi_{mp}^{\mathrm{WP}}\xi_{ms}^{\mathrm{WP}}\xi_{nu}^{\mathrm{WP}}\rangle = \sigma_{\mathrm{WP}}^4\left(\delta_{in}\delta_{ku}\delta_{ps} + \delta_{im}\delta_{mn}(\delta_{kp}\delta_{su}+\delta_{ks}\delta_{pu}\right) \tag{S140}$$

such that

$$\begin{aligned}\tfrac{1}{2}\langle \mathrm{tr}[\Delta w S^\mu \Delta w^T]\rangle &= \tfrac{1}{2}\eta^2 M\,\mathrm{tr}[WSSW^T]\cdot\mathrm{tr}[SS^\mu S] + \eta^2\,\mathrm{tr}[WSSS^\mu SSW^T] \\ &= \eta^2 M\alpha_\mu^6\cdot\sum_{\nu=1}^{N}\alpha_\nu^2\langle E^\nu(n)\rangle + 2\eta^2\alpha_\mu^8\langle E^\mu(n)\rangle.\end{aligned} \tag{S141}$$

With this, the evolution of the expected error is described by

$$\langle E^\mu(n+1)\rangle = \sum_{\nu=1}^{N}(A+B)_{\mu\nu}\langle E^\nu(n)\rangle, \tag{S142}$$

where

$$A_{\mu\nu} = (1-2\eta\alpha_\mu^4+\eta^2\alpha_\mu^8)\delta_{\mu\nu}, \tag{S143}$$

$$B_{\mu\nu} = \eta^2 M\alpha_\mu^2\alpha_\nu^6 + \eta^2\alpha_\mu^8\delta_{\mu\nu}. \tag{S144}$$

For the case of repeating inputs and where N_{eff} eigenvalues of S are equal to α^2 and all others zero, the expected error evolves as

$$\langle E(n+1)\rangle = \left(\langle E(n)\rangle - E_{\mathrm{opt}}\right)\cdot a + E_{\mathrm{opt}}, \qquad a = 1-2\eta\alpha^4+\eta^2\alpha^8(MN_{\mathrm{eff}}+2). \tag{S145}$$

Minimizing a leads to an optimal learning rate and convergence factor of

$$\eta^* = \frac{1}{(MN_{\text{eff}}+2)\alpha^4}, \qquad a^* = 1 - \frac{1}{MN_{\text{eff}}+2}. \tag{S146}$$

In this case, HP converges as fast as both WP and NP. In addition, irrelevant weights do not diverge, which benefits the learning of multiple subtasks. Further we expect the final error to be lower than for NP because all output perturbations lie in the realizable subspace, which is confirmed by our numerical simulations, Fig. 4. Thus for latent inputs that have the same strength, HP combines the advantages of both WP and NP. Note that, as for WP and NP but in contrast to WP0, we can without loss of generality change to a set of rotated inputs, because weight perturbations are isotropic such that the update equation Eq. (S134) is invariant under the rotation. This is also reflected by Eqs. (S142–S144): the equations show that the error evolution only depends on the eigenvalues α_μ^2 of S, which are invariant to input rotations.

We observed that application of HP to the reservoir computing task gave worse performance than application of WP and NP (main text, Sec. "Conclusions from the theoretical analysis and new learning rules"). We explain this by the fact that HP generates biased updates for latent inputs with different strengths (a similar explanation may hold for the unsatisfactory performance on MNIST): weights connected to stronger input components both have a larger impact on output perturbations, as for WP, and get updated more strongly due to their eligibility traces being larger, as for NP (compare Eqs. (S108,S109)). The mean update is thus biased,

$$\begin{aligned}\langle \Delta w_{ij}^{\text{HP}} \rangle &= -\frac{\eta}{\sigma_{\text{WP}}^2} \left\langle \text{tr}[W S \xi^{\text{WP}T}] \sum_{k=1}^{N} \xi_{ik}^{\text{WP}} S_{kj} \right\rangle = -\frac{\eta}{\sigma_{\text{WP}}^2} \sum_{m=1}^{M} \sum_{klp=1}^{N} W_{ml} S_{lp} S_{kj} \langle \xi_{mp}^{\text{WP}} \xi_{ik}^{\text{WP}} \rangle \\ &= -\eta \sum_{i=1}^{M} \sum_{jk=1}^{N} W_{il} S_{lk} S_{kj} = -\eta \sum_{k=1}^{N} \frac{\partial E}{\partial w_{ik}} S_{kj}.\end{aligned} \tag{S147}$$

If inputs are rotated so that $w_{i\mu}$ reads out from the μth input component $r_{\mu t}$ of strength α_μ^2, then the mean updates $\langle \Delta w_{i\mu}^{\text{HP}} \rangle = -\eta \frac{\partial E}{\partial w_{i\mu}} \cdot \alpha_\mu^2$ are proportional to their weight error gradients multiplied by α_μ^2. This means that weights related to weak inputs are updated only little and will take a long time to converge. If such weak inputs are important to realize the target, HP will perform worse than NP and WP. Adding a network layer that equalizes the non-zero (or non-negligible) input strengths in each subtask might render HP beneficial and applicable.

Recovering unbiased updates in presence of inhomogeneous input strengths by multiplying updates with the inverse correlation matrix (which is a non-local operation), $\Delta w^{\text{HP}} \to \Delta w^{\text{HP}} S^{-1}$, simply reduces HP to WP,

$$\begin{aligned}\Delta w_{ij}^{\text{HP,unbiased}} &= -\frac{\eta}{\sigma_{\text{WP}}^2 T}(E^{\text{pert}} - E) \sum_{l=1}^{N} \sum_{t=1}^{T} \xi_{it}^{\text{out}} r_{lt} \cdot S_{lj}^{-1} = -\frac{\eta}{\sigma_{\text{WP}}^2 T}(E^{\text{pert}} - E) \sum_{t=1}^{T} \sum_{kl=1}^{N} \xi_{ik}^{\text{WP}} r_{kt} r_{lt} S_{lj}^{-1} \\ &= -\frac{\eta}{\sigma_{\text{WP}}^2}(E^{\text{pert}} - E) \sum_{kl=1}^{N} \xi_{ik}^{\text{WP}} S_{kl} S_{lj}^{-1} = -\frac{\eta}{\sigma_{\text{WP}}^2}(E^{\text{pert}} - E) \xi_{ij}^{\text{WP}} = \Delta w_{ij}^{\text{WP}},\end{aligned} \tag{S148}$$

as long as all input strengths are non-zero such that S^{-1} is well-defined. Setting the eigenvalues of S^{-1} related to zero inputs to zero (a priori they are undefined) means that irrelevant weight combinations are not updated, which for rotated inputs reduces the unbiased version of HP to WP0.

Figures and Data

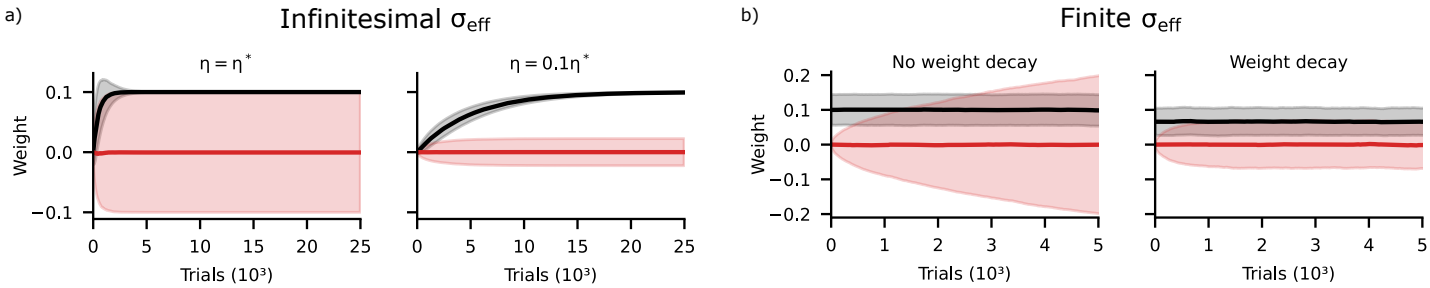

Figure S1. Further analysis of the weight diffusion in WP. a) Diffusion of irrelevant weights is transient for infinitesimal perturbation size. Display like main text, Fig. 2b, but for infinitesimal σ_{WP}. The relevant weights converge to the relevant weights of the teacher network, $w^*_{\mathrm{rel},i} = 0.1$. Learning with optimal rate ($\eta = \eta^*$) leads to a final standard deviation of irrelevant weights of the same size as the mean relevant weights (left); a smaller learning rate leads to less weight diffusion (right). SM2, Sec. "Transient weight diffusion due to credit assignment-related update fluctuations" explains these observations. b) Weight diffusion for finite perturbation size, after the output error has decayed to its stationary residual value and the relevant weights fluctuate around their targets. The irrelevant weights, which are initially set to zero, diffuse without bounds (left). In particular, the standard deviation of the weight distribution grows like $\sim \sqrt{n}$ with the learning trial number n. A tendency of the weights to decay confines this growth (right).

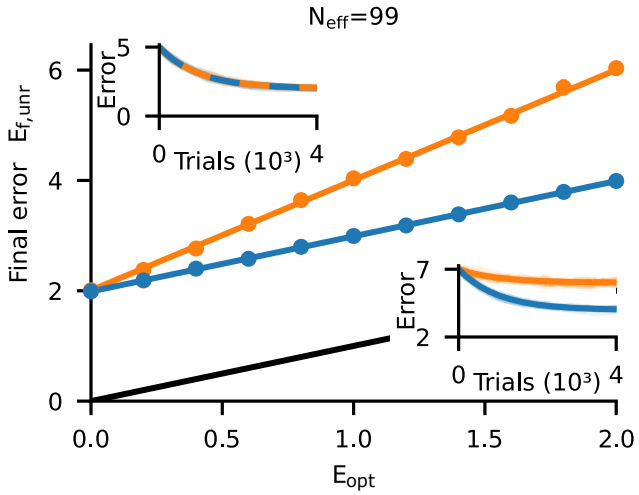

Figure S2. Final error after convergence as a function of the limiting error E_{opt} like Fig. 3b, but for effective input dimensionality $N_{\text{eff}} = 99$ (instead of $N_{\text{eff}} = 50$). Since $T = 100$ and $N_{\text{eff}} = 99$ (N_{eff} must be smaller than T to allow an unrealizable part), both algorithms perform basically the same for $E_{\text{opt}} = 0$ (cf. also light curves in main text, Fig. 1b, left). For $E_{\text{opt}} > 0$ the final error of WP is shifted by E_{opt} while that of NP increases approximately by $2E_{\text{opt}}$.

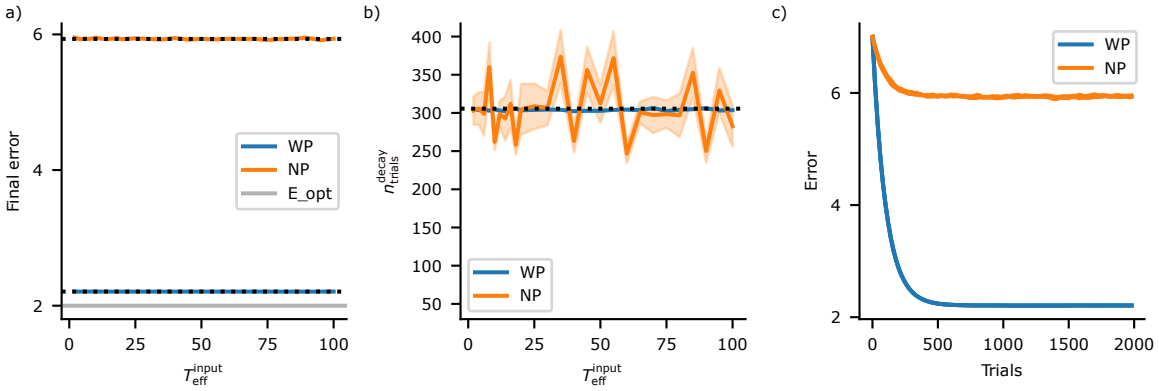

Figure S3. WP and NP are unaffected by input correlations. a) Final error of WP (blue) and NP (orange) in a task where the inputs are temporally correlated, versus the effective temporal dimension $T_{\mathrm{eff}}^{\mathrm{input}}$ of the inputs. For $T_{\mathrm{eff}}^{\mathrm{input}} = T = 100$ we have uncorrelated input, for $T_{\mathrm{eff}}^{\mathrm{input}} = 2$ the input traces vary very slowly. Inputs are generated by orthonormalizing exponentially filtered white noise. Realizable target components are linear combinations of the correlated inputs. We assume that there is an additional unrealizable target component, which, for simplicity, contains all modes orthogonal to the inputs with equal strength. Our theoretical results (black dotted curves) predict that the final error is independent of the correlation time for both WP and NP. This is confirmed by the numerical simulations (colored curves, overlayed by the theoretical ones). b) Number of trials to reach 95% of the final error reduction for WP (blue) and NP (orange) as a function of $T_{\mathrm{eff}}^{\mathrm{input}}$. Our theoretical results (black dotted curves, overlapping) predict that the decay time of the expected error is independent of the correlation time and the same for both WP and NP. This is confirmed by the numerical simulations (mean: colored curves, partially underlaying the theoretical ones, shaded: standard error of the mean). Since the learning curves of NP are more variable, its mean convergence time has a higher standard error. c) Mean error (solid) and standard error of the mean error (shaded) as a function of trial number for different input correlation times (effective temporal input dimensions: $T_{\mathrm{eff}}^{\mathrm{input}} = 2, 10, 100$). The curves agree within their errors and are not visually distinguishable. Parameters: $M = N_{\mathrm{eff}} = 10$, $T = 100$, $\sigma_{\mathrm{eff}} = 0.04$, $E_{\mathrm{opt}} = 2$. $T_{\mathrm{eff}}^{\mathrm{input}}$ is in (a,b) varied from 2 to 100 in steps of 2 between 2 and 20 and in steps of 5 between 25 and 100, with 1000 repetitions each. The final error in (a) is measured after 2000 trials.

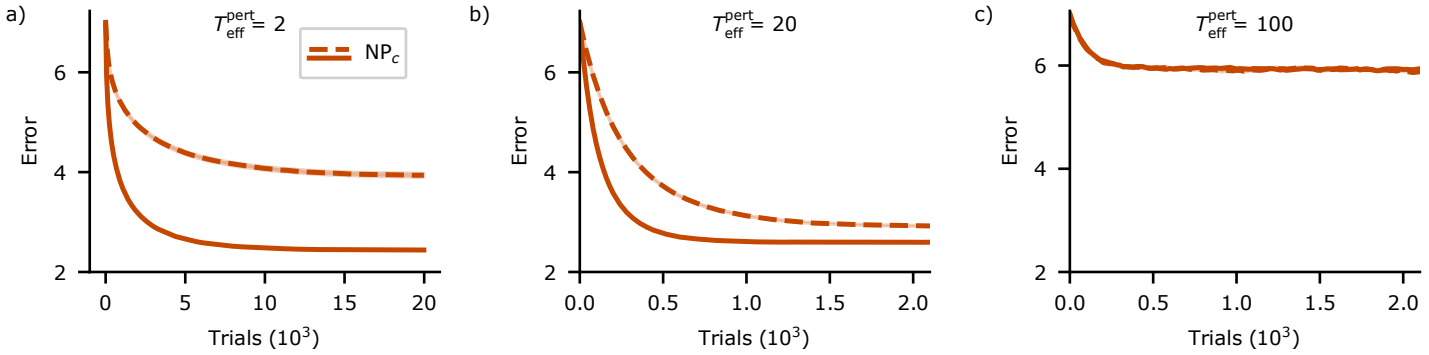

Figure S4. Exemplary error decay curves of NPc (mean and SEM) for different input and perturbation correlation times. a) For $T_{\text{eff}}^{\text{pert}} = 2$, that is for larger perturbation than input correlation time, convergence slows down (note the changed x-axis scale compared to b,c). For correlated inputs (solid) but not for uncorrelated inputs (dashed), NPc still achieves a low final error. b) For $T_{\text{eff}}^{\text{pert}} = 20$, the perturbations of NPc have the same effective temporal dimension as the correlated inputs (solid curves). NPc simultaneously achieves a low final error and fast convergence. c) For $T_{\text{eff}}^{\text{pert}} = 100 = T$, NPc reduces to NP and settles at the same high final error. As NP is insensitive to input correlations (Fig. S3), the curves of NPc for correlated and uncorrelated inputs here agree. Parameters: $N_{\text{eff}} = M = 10$, $N = T = 100$, $\sigma_{\text{eff}} = 0.04$, $E_{\text{opt}} = 2$. Latent inputs of equal strength are constructed by orthonormalizing white noise (red dashed) or exponentially filtered white noise (with time constant $\tau_{\text{corr}}^{\text{input}} = 4$ and thus $T_{\text{eff}}^{\text{input}} = 20$, red solid). Unrealizable target components are constructed from the last $T - N_{\text{eff}}$ orthonormalized noise traces with equal strengths.

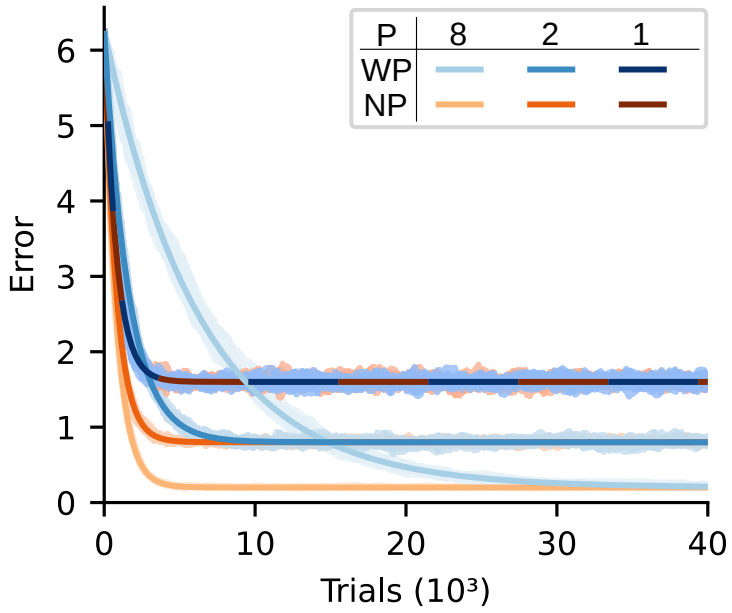

Figure S5. Error dynamics scale differently for WP and NP when splitting the input into different patterns for different trials. The network is the same as the one used in Fig. 1 ($N = 100$, $M = 10$, $\sigma_{\text{eff}} = 4 \times 10^{-2}$). The task is to reproduce the output of a teacher network in response to input with a dimensionality of $N_{\text{eff}}^{\text{task}} = 80$, which we split into P non-overlapping input patterns each having dimensionality $N_{\text{eff}}^{\text{trial}} = N_{\text{eff}}^{\text{task}}/P$ (hence $PN_{\text{eff}}^{\text{trial}} = 80$). For simplicity, we use input patterns where at each timestep a different input unit has the value $\sqrt{N/N_{\text{eff}}^{\text{task}}}$, while all other input units are zero. This implies $T = N_{\text{eff}}^{\text{trial}}$. The figure shows error curves for WP (blue) and NP (orange) from simulations (10 runs, shaded) together with analytical curves for the decay of the expected error (solid), for different values of P (simulation results for NP with $P = 8$ are mostly covered by the analytical curve). Theoretical curves and simulations agree well. The convergence speed of WP decreases with increasing P (Eqs. (S78,S92)). The convergence speed of NP is almost unaffected by P for NP (Eqs. (S79,S93)). The residual error is inversely proportional to P for both WP and NP (Eqs. (S82,S83)) and it is equal for WP and NP because $T = N_{\text{eff}}^{\text{trial}}$.

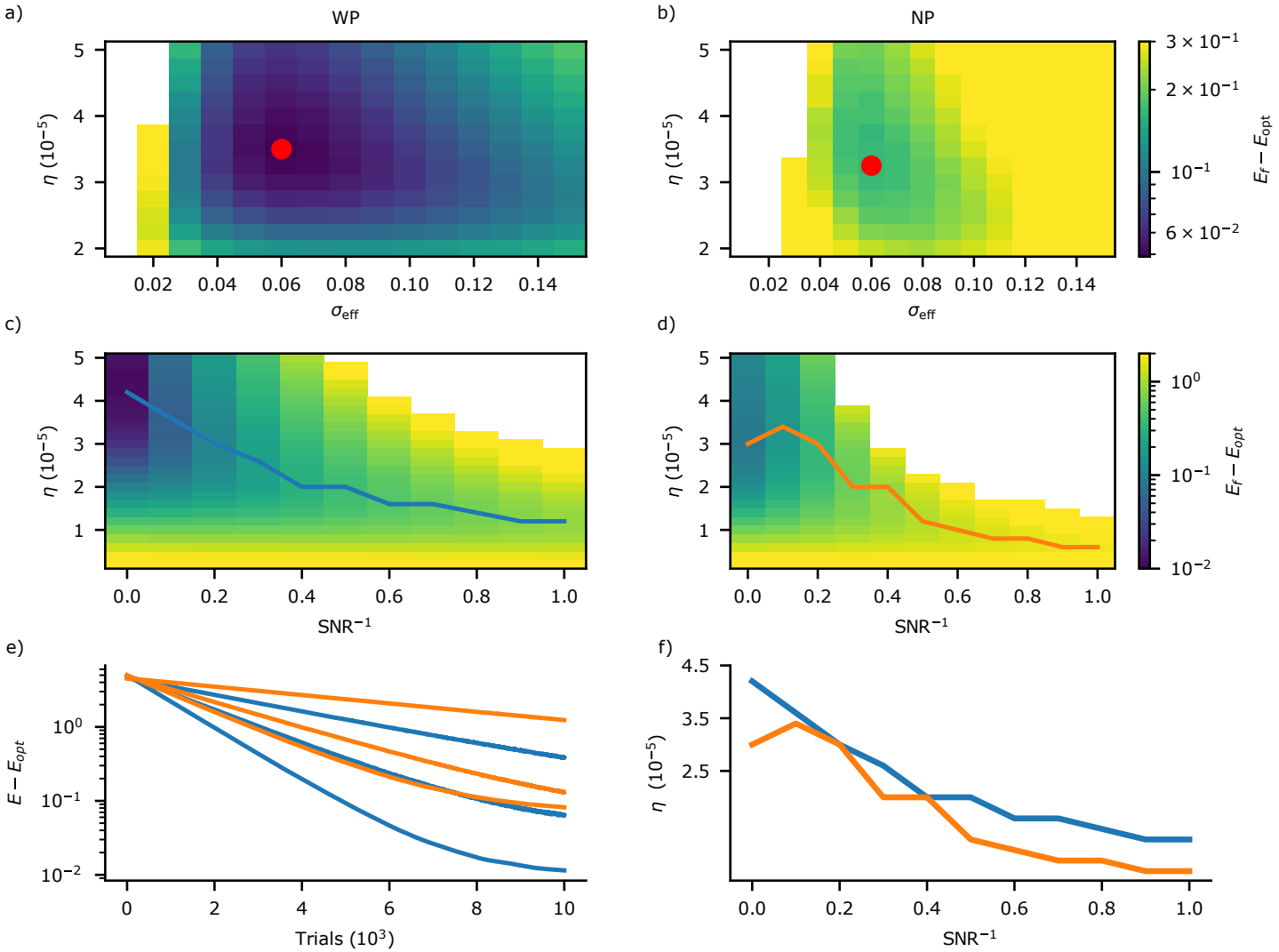

Figure S6. Optimal perturbation size and optimal learning rate for different levels of input noise. a,b) Excess of the final error of WP (a) and NP (b) beyond E_{opt}, for input noise with $\text{SNR}^{-1} = 0.1$ and different perturbation strengths and learning rates. The white region corresponds to parameters for which the algorithms did not converge. The red dots indicate the optimal parameter combinations. They have the same $\sigma_{\text{eff}} = 0.06$ for WP and NP. WP can afford a higher learning rate and reaches a lower minimal error. Further it converges on a larger parameter region. c,d) Final error of WP (c) and NP (d) beyond E_{opt}, for $\sigma_{\text{eff}} = 0.06$ and different input noise strengths and learning rates. The optimal learning rates, which yield the lowest error, are highlighted (WP: blue connected points, NP: orange). e) Exemplary evolution (mean and SEM) of the error beyond E_{opt}, for WP (blue) and NP (orange) and $\text{SNR}^{-1} \in \{0, 0.3, 1\}$ (bottom to top curves). The learning rates are set to the optimal values. f) Joint depiction of the optimal learning rates of WP and NP, from (c) and (d). These are also the learning rates used in main text, Fig. 6. Parameters: $M = N_{\text{eff}} = 10$, $N = T = 100$, $\alpha^2 = N/N_{\text{eff}} = 10$, $\sigma_{\text{eff}} = 0.04$, $d = 0$. The best achievable error E_{opt} is in presence of input noise nonzero despite $d = 0$, because the noise prevents an exact reproduction of the target. SNR is defined as the ratio of the total (summed) power in the input signal to that in the noise. Averages are taken over the last 1000 of $10\,000$ trials and 100 repetitions. e) shows mean and SEM.

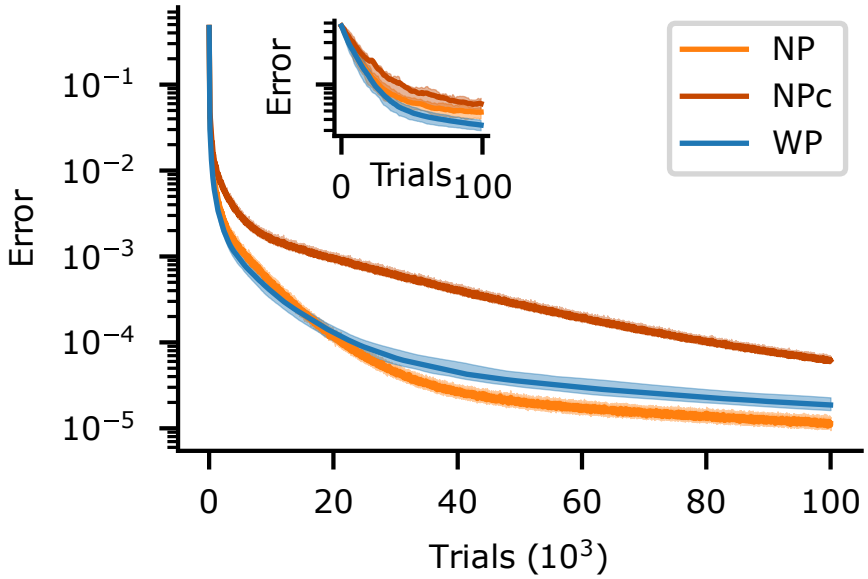

Figure S7. Error dynamics of the reservoir-based drawing task like Fig. 7c, with the same parameters but for infinitesimally small perturbation size σ_{eff}. WP and NP perform similarly, NP slightly better. NPc performs worse, indicating that the optimal learning rate needs to be adapted when changing $T_{\text{eff}}^{\text{pert}}$, like in Fig. 4 (see main text "Materials and Methods"). The error curves of WP and NP differ because their error components interfere differently (SM4, Sec. "Evolution of error components related to strong and weak inputs"). Depending on the initial weight mismatch, either WP or NP can show the faster initial improvements. NP converges faster towards the end of training. Simulations suggest that the asymptotic ratio of the convergence rates of NP and WP is, however, only on the order of 1, Fig. S9. The curves show median (solid) and interquartile range (shaded) computed over 100 repetitions.

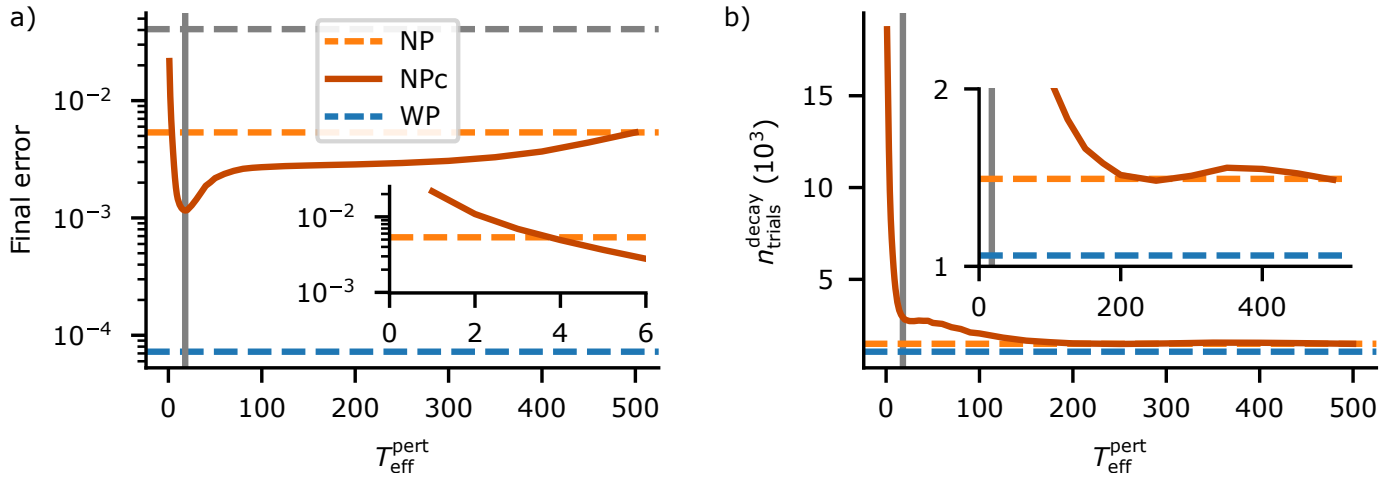

Figure S8. Final error and convergence time of NPc applied to the task in main text, "Reservoir computing-based drawing task", for $\eta_{\text{NPc}} = \eta_{\text{NP}}^*$ and different correlation times of the perturbations. We note that in contrast to Fig. 4 the learning rate is not adapted when changing T_{eff}, see main text "Materials and Methods". a) Final error of NPc (red) as a function of $T_{\text{eff}}^{\text{pert}}$. For $T_{\text{eff}}^{\text{pert}} \geq 4$ (see also inset), NPc achieves a final error equal to or lower than NP (orange dashed), but not as low as WP (blue dashed). The effective temporal perturbation dimension $T_{\text{eff}}^{\text{pert,opt}} = 18$, which minimizes the final error, is marked by a vertical line. The error achieved by a least squares fit using only the largest 5 principle components of the reservoir is shown for comparison (gray dashed). The NPc curve shows mean and SEM of the final error over 1000 repetitions and the last 1000 of $30\,000$ trials. b) The number of trials needed to achieve 99% of the final error reduction, $n_{\text{trials}}^{\text{decay}}$, stays largely constant for $T_{\text{eff}} > 200$ (inset) and increases slightly for $T_{\text{eff}}^{\text{pert,opt}} \leq T_{\text{eff}}^{\text{pert}} < 200$. Reducing $T_{\text{eff}}^{\text{pert}}$ below $T_{\text{eff}}^{\text{pert,opt}}$ strongly increases $n_{\text{trials}}^{\text{decay}}$. Results for NP (orange dashed) and WP (blue dashed) are shown for comparison. To determine $n_{\text{trials}}^{\text{decay}}$ we consider 10 samples of 100 runs each. For each sample, the mean error over runs is computed and additionally smoothed with a centered temporal running average of window size 100. $n_{\text{trials}}^{\text{decay}}$ is then the trial at which the described average drops for the first time below $E_{f,\text{unr}} + 0.01 \cdot \left(E(0) - E_{f,\text{unr}}\right)$. b) reports the mean and SEM of $n_{\text{trials}}^{\text{decay}}$ over samples.

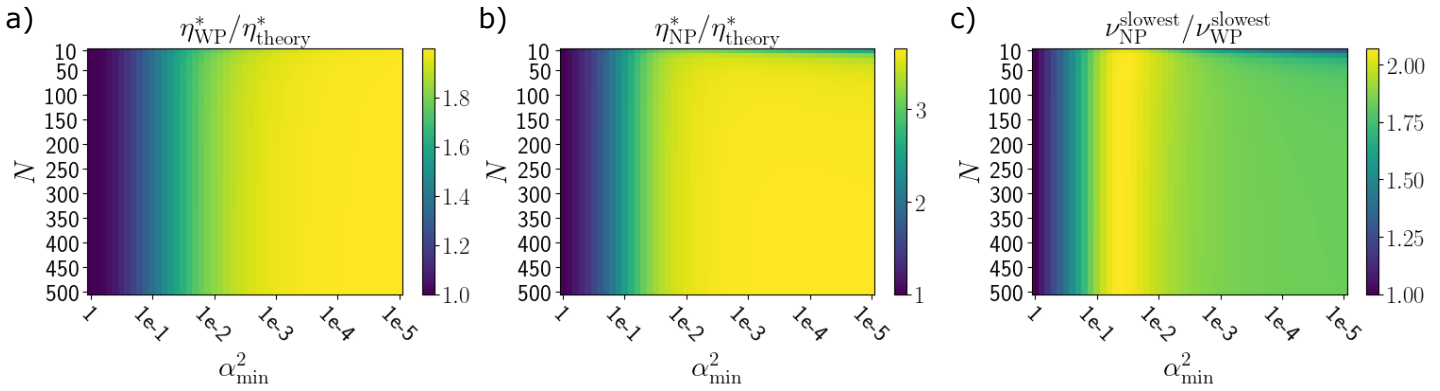

Figure S9. Learning rates and convergence speed of WP and NP when learning a single input-output pairing, for input strength distributions of different widths. (a,b) An analytical estimate η^*_{theory} of the optimal learning rate based on the participation ratio PR (main text, "Materials and Methods") broadly agrees with semi-analytical results $\eta^*_{\mathrm{WP|NP}}$ that maximize the convergence speed of the slowest decaying error mode (Eqs. (S103,S112)). c) The convergence rates $\nu^{\mathrm{slowest}}_{\mathrm{WP}}$ and $\nu^{\mathrm{slowest}}_{\mathrm{NP}}$ of the slowest decaying error component of WP and NP at optimal learning rates $\eta^*_{\mathrm{WP|NP}}$ stay comparable even when the input strengths differ by orders of magnitude. For each combination of the hyperparameters N and α^2_{min}, N orthogonal input components are constructed with exponentially decaying strengths $\alpha^2_\mu = \alpha^2_0 \cdot \gamma^\mu$ where $\gamma \leq 1$ is chosen such that $\alpha^2_0 = 1$ and $\alpha^2_N = \alpha^2_{\mathrm{min}}$. Here $\alpha^2_{\mathrm{min}} = 1$ reproduces the theory case with $\alpha^2 = 1$ and $N_{\mathrm{eff}} = N$, whereas $\alpha_{\mathrm{min}} = 1 \times 10^{-5}$ corresponds to a broad input strength distribution with $\mathrm{PR} \ll N$. N and α^2_{min} are varied on a 50×51 grid. In (a,b) we compute for each pair of hyperparameters the participation ratio PR from the distribution of input strengths and employ it to predict the optimal learning rate $\eta^*_{\mathrm{theory}} = 1/(M\mathrm{PR} + 2)\overline{\alpha^2}$. Here $M = 10$ and $\overline{\alpha^2} = \sum_{\mu=1}^{N} \alpha^2_\mu/\mathrm{PR}$ is the mean input strength per effective input dimension. For the comparison, we construct the matrices of error evolution $(A + B)_{\mathrm{WP|NP}}[\eta]$ (Eqs. (S107–S109)) and compute the optimal learning rate for the slowest component, $\eta^*_{\mathrm{WP|NP}}$, by numerically minimizing the largest eigenvalue of $(A + B)_{\mathrm{WP|NP}}[\eta]$. (c) displays $\nu_{\mathrm{WP|NP}} \approx 1 - a_{\mathrm{WP|NP}}$, where $a_{\mathrm{WP|NP}}$ is the largest eigenvalue of $(A + B)_{\mathrm{WP|NP}}[\eta^*]$.

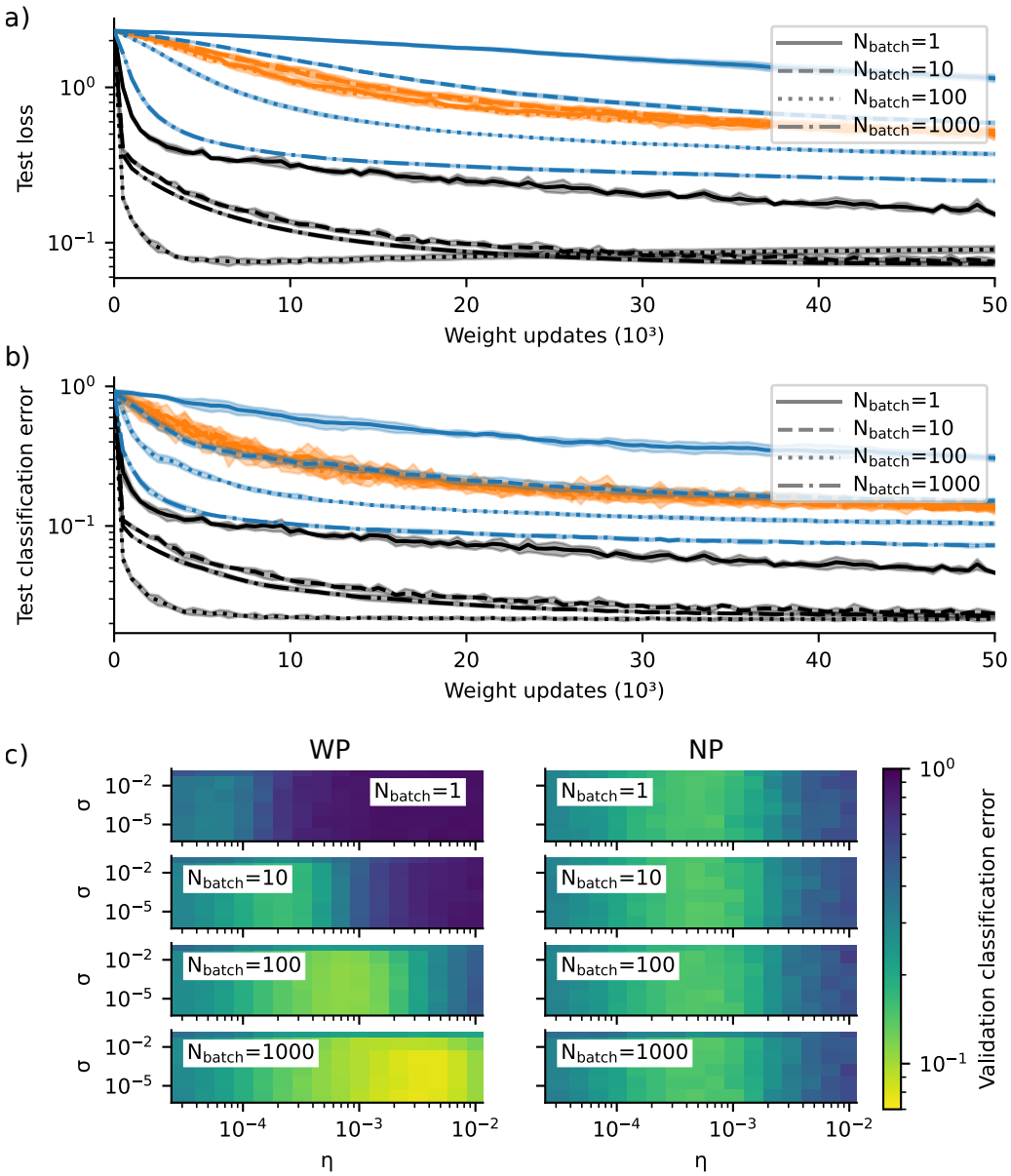

Figure S10. Test loss, test accuracy and grid search results for the MNIST task. a) Test loss (cross entropy loss) for the best parameters found in the grid search for WP (blue), NP (orange) and SGD (black). We note that the obtained optimal learning rate for SGD and $N_{\text{batch}} = 100$ is comparably large (Tab. 2); SGD therefore seems to overfit slightly. Lines show the mean and shaded areas show the standard deviation using 5 network instances. b) Same as a) but for the test classification error (one minus test accuracy). The grid search yields for SGD and $N_{\text{batch}} \geq 100$ similar final accuracies for very different learning rates. Therefore the best learning rates, which maximize the final accuracies, and thus also the learning curves shown here, can in this case be quite different from each other. c) Grid search to estimate the optimal learning rates for WP and NP with different perturbation strengths and batch sizes. The figure displays the mean classification error after 50000 weight updates, for 5 instances, on a validation data set not used for training.

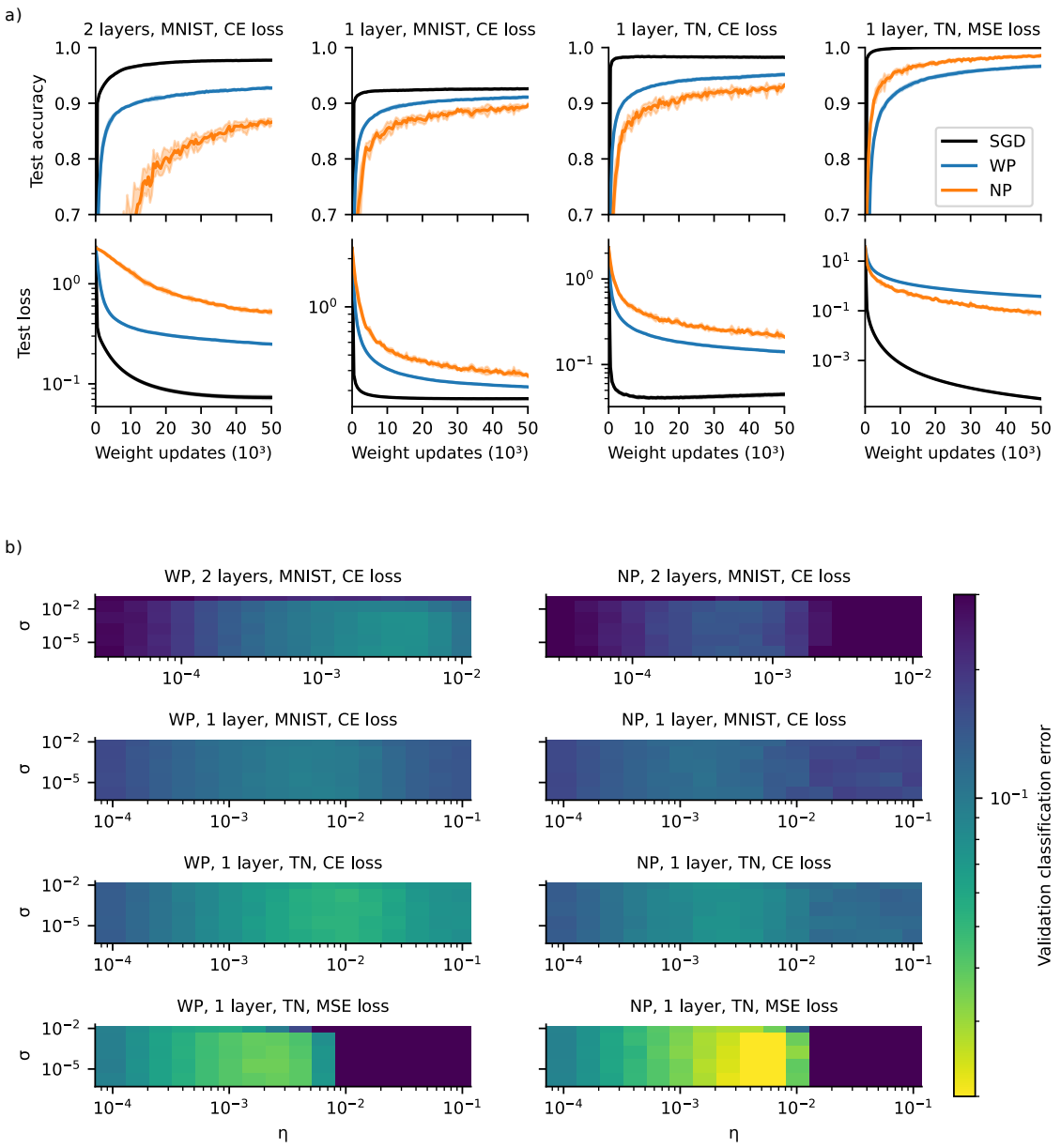

Figure S11. Learning performance and grid search results for four variations of the MNIST task. Specifically, the panels show learning in a two-layer network trained on MNIST using cross-entropy loss (2 layers, MNIST, CE loss; same as in Fig. 9b and Fig. S10), a single-layer network trained the same way (1 layer, MNIST, CE loss), a single-layer network with the MNIST images as input but with target labels determined by the maximal output of a teacher network that was trained on MNIST using SGD (1 layer, TN, CE loss) and a single-layer linear network with the MNIST images as input using mean-squared error loss with targets given by the raw output of the same teacher network (1 layer, TN, MSE loss). For the single-layer networks we simply remove the hidden layer from the two-layer network and in the last case (1 layer, TN, MSE loss) also remove the softmax-nonlinearity from the output layer. This yields a single layer linear network with realizable targets. We only consider $N_{\text{batch}} = 1000$ and perform a grid search to find the best performing learning parameters.
a) Test accuracy (upper row) and loss (lower row) for WP (blue), NP (orange) and SGD (black) for the best performing learning parameters. Removing the hidden layer worsens the performance of WP and SGD but improves it for NP (compare first to second column). Using a teacher network to create the target labels does not change the relative performance of WP and NP, indicating that in this task unrealizable target labels, e.g. due to bad handwriting, do not significantly harm NP (compare third to second column). Note that this does not mean that the targets are exactly realizable, because it is not possible for the network to reproduce the binary target output given by the one-hot encoded targets. Further removing all nonlinearities from the networks and using mean-squared error loss leads to better performance of NP compared to WP (compare fourth to third column). In the case of training using a teacher network, we determine the accuracy by using the index of the maximal output of the teacher network as the target label. Solid lines show the mean and shaded areas the standard deviation using 5 network instances. b) Grid search results for WP and NP as given by the mean classification error for 5 instances on a validation data set not used for training. The error is clipped at 0.3 for better visualization.

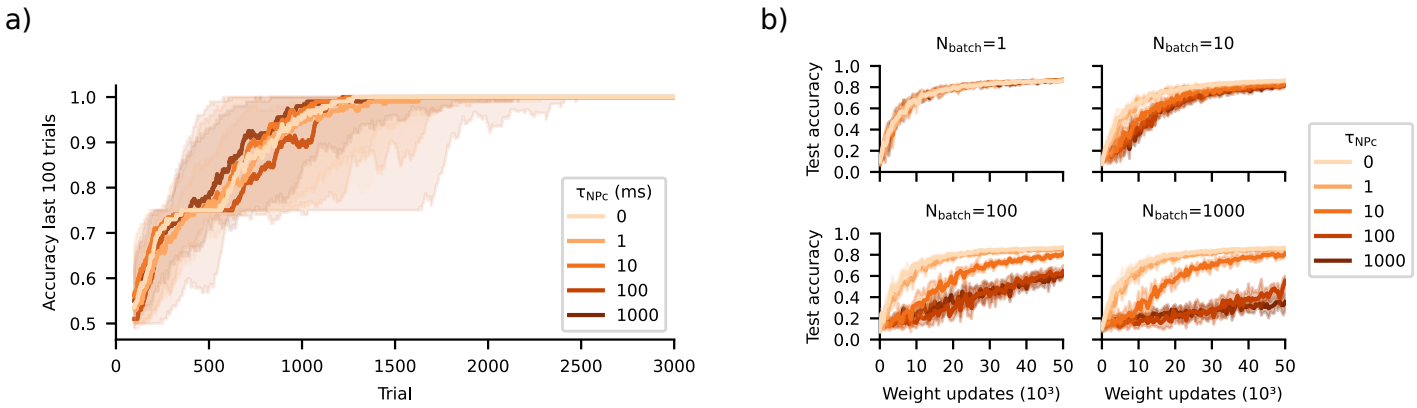

Figure S12. Time-correlated NP (NPc) does not improve task performance for the DNMS task and MNIST. a) Same as Fig. 8c but for NP ($\tau_{\mathrm{NPc}} = 0\,\mathrm{ms}$) and NPc with different filtering time constants ($\tau_{\mathrm{NPc}} = 1\,\mathrm{ms},\ 10\,\mathrm{ms},\ 100\,\mathrm{ms}$ and $1000\,\mathrm{ms}$). NPc does not improve task performance. b) Same as Fig. 9b but for NP ($\tau_{\mathrm{NPc}} = 0$) and NPc with different filtering time constants ($\tau_{\mathrm{NPc}} = 1,\ 10,\ 100$ and 1000). For $N_{\mathrm{batch}} = 1$, NP and NPc are the same learning rule because trials are not temporally extended, i.e. $T = N_{\mathrm{batch}} = 1$. For larger values of N_{batch}, NPc worsens with increasing filtering time constant. This is because the inputs in each trial are random sequences of images, whose pixel values are uncorrelated in time. Further, for a given filtering time constant $\tau_{\mathrm{NPc}} > 0$, NPc worsens with increasing batch size N_{batch}. This may be because smaller batches are more likely to contain similar images of only a few numbers, for which learning with near-constant node perturbations still works.

Algorithm	N_{batch}	η	Test loss	Test accuracy
WP	1	6.81×10^{-5}	1.160(55)	0.690(20)
	10	2.15×10^{-4}	0.613(15)	0.839(9)
	100	6.81×10^{-4}	0.390(8)	0.890(3)
	1000	3.16×10^{-3}	0.270(7)	0.923(2)
NP	1	6.81×10^{-4}	0.515(26)	0.856(7)
	10	4.64×10^{-4}	0.541(19)	0.860(11)
	100	6.81×10^{-4}	0.510(36)	0.860(14)
	1000	4.64×10^{-4}	0.545(25)	0.859(5)
SGD	1	0.010	0.165(7)	0.952(3)
	10	0.056	0.083(6)	0.976(1)
	100	0.562	0.098(7)	0.977(1)
	1000	0.056	0.079(7)	0.977(2)

Table 2. Network performance for SGD, WP and NP on a held-out test set, after training, for the MNIST task. The third column shows the best learning rate obtained from the grid search. Values in the last two columns are the mean loss and the accuracy after 50000 weight updates, averaged over five instances (standard deviation in brackets).

APPENDIX C

REINFORCE and weight perturbations

When node activations fluctuate randomly, the REINFORCE framework yields NP updates [14], Eq. (2.29). Does the REINFORCE framework then yield WP updates for networks with randomly fluctuating weights?

To test this, consider (naively, as explained below) applying the episodic REINFORCE update rule, Eq. (2.26), to neurons that have stochastic outputs due to perturbations of their weights. Concretely, the node activations are given by Eq. (2.23) (left), with iid. Gaussian-distributed weight perturbations ξ_{ij}^{WP} with mean zero and variance σ_{WP}^2. Then the *individual* node activations z_{it}^{pert} are also Gaussian-distributed, with probability density

$$f(z_{it}^{\mathrm{pert}}, \mathbf{w}, \mathbf{r}) = g\Big(z_{it}^{\mathrm{pert}},\ \mu = \sum_j w_{ij} r_{jt},\ \sigma^2 = \sum_j r_{jt}^2 \sigma_{\mathrm{WP}}^2\Big). \tag{C.1}$$

Consider the case where all node activations have the same variance $\sum_j r_{jt}^2 \sigma_{\mathrm{WP}}^2 \equiv \sigma_{\mathrm{out}}^2$ and define $\xi_{it}^{\mathrm{out}} = z_{it}^{\mathrm{pert}} - z_{it}$. Then, by comparing ξ^{out} with ξ^{NP}, we see that the stochastic units (*individually*) behave just like for NP, with σ_{out}^2 corresponding to σ_{NP}^2. Consequently, the REINFORCE framework yields 'NP updates' also for the case where the weights fluctuate.

In publication 1 [1], this learning rule is coined 'Hybrid Perturbation' (HP), because it combines weight perturbations with NP updates:

$$z_{it}^{\mathrm{pert}} \overset{\mathrm{HP}}{=} \sum_{j=1}^{N} (w_{ij} + \xi_{ij}^{\mathrm{WP}}) r_{jt}, \qquad \Delta w_{ij}^{\mathrm{HP}} = -\frac{\eta}{\sigma_z^2} (E^{\mathrm{pert}} - E) \sum_{t=1}^{T} \xi_{it}^{\mathrm{out}} r_{jt}. \tag{C.2}$$

This is motivated by the observation that the node fluctuations caused by weight perturbations are, through their input-dependence, more relevant. On the other hand, the single backpropagation step in the NP rule, i.e., from the node gradient estimate to the weight gradient by the multiplication with r_{jt} in the eligibility trace, solves part of the credit assignment problem. A combination of the two mechanisms thus seems promising.

However, publication 1 [1] shows that HP updates are not parallel to the weight gradient of the error,

but biased. Define the input correlation matrix by

$$S_{ij} \equiv \frac{1}{T}\sum_{t=1}^{T} r_{it} r_{jt}. \tag{C.3}$$

Then the expected HP update is (see supplement 1 [120], eq. S147)

$$\langle \Delta w_{ij}^{\mathrm{HP}} \rangle = -\eta \sum_{k=1}^{N} \frac{\partial E}{\partial w_{ik}} S_{kj}. \tag{C.4}$$

Clearly, when S is not a multiple of the identity matrix, the HP updates will be misaligned with the weight gradient. Why is that? Consider the case where S is diagonal, but with different entries, describing orthogonal inputs with different strengths $\alpha_i^2 = (1/T)\sum_{t=1}^{T} r_{it}^2$. Then the perturbations of weights mediating strong inputs contribute strongly to the output fluctuations ξ_{it}^{out}, while weights that mediate weak inputs contribute only weakly. Consequently, the part of the error signal $E^{\mathrm{pert}} - E$ that is related to the perturbation of a weight, and contributes systematically to its update, is larger for strong-input-mediating weights. Up to this point, correlating the error signal with the weight perturbations as in WP would lead to unbiased updates, because, after all, the weight gradient is also larger for weights with strong inputs. HP, however, applies NP updates instead. In the computation of the eligibility traces, the weight updates are then *again* scaled with the (square root of the) input strength, through the multiplication with r_{jt} in Eq. (2.24).

The update bias is surprising when considering that REINFORCE updates are proven to follow the gradient on average [29], see Eq. (2.28), and HP can be derived from the REINFORCE framework, see above. Is there a contradiction? The answer is that, for temporally extended tasks, weight perturbations induce temporally-correlated node perturbations:

$$\langle \xi_{it}^{\mathrm{out}} \xi_{is}^{\mathrm{out}} \rangle = \sum_{jk=1}^{N} \underbrace{\langle \xi_{ij}^{\mathrm{WP}} \xi_{ik}^{\mathrm{WP}} \rangle}_{=\sigma_{\mathrm{WP}}^2 \delta_{jk}} r_{jt} r_{ks} = \sigma_{\mathrm{WP}}^2 \cdot \sum_{j=1}^{N} r_{jt} r_{js}. \tag{C.5}$$

The REINFORCE framework [29], however, (implicitly) assumes that the stochastic activation z_{it}^{pert} of node i depends only on the weights w_{ij} presynaptic to it and its inputs r_{jt}; in particular, it does not depend on the (perturbed) activations of other nodes or its own previous or later activation. The REINFORCE framework is therefore not applicable to networks experiencing correlated fluctuations.

Interestingly, if HP updates are decorrelated by multiplication with the inverse input correlation matrix S^{-1}, given its existence, the HP rule reduces to WP, see ref. [120] eq. S148.

APPENDIX D

Publication 2: Cooperative coding of continuous variables in networks with sparsity constraint

This appendix, 'publication 2' [2], contains a full copy of the following article (version 2):

[2] **P. Züge** and R.-M. Memmesheimer
Cooperative Coding of Continuous Variables in Networks with Sparsity Constraint
bioRxiv (2024):2024.05.13.593810v2

For the contribution statement, see Chapter 4.

Cooperative coding of continuous variables in networks with sparsity constraint

Paul Züge•, Raoul-Martin Memmesheimer*

Institute for Genetics, University of Bonn, Germany

* rm.memmesheimer@uni-bonn.de, • pzuege@uni-bonn.de

Abstract

A hallmark of biological and artificial neural networks is that neurons tile the range of continuous sensory inputs and intrinsic variables with overlapping responses. It is characteristic for the underlying recurrent connectivity in the cortex that neurons with similar tuning predominantly excite each other. The reason for such an architecture is not clear. Using an analytically tractable model, we show that it can naturally arise from a cooperative coding scheme. In this scheme neurons with similar responses specifically support each other by sharing their computations to obtain the desired population code. This sharing allows each neuron to effectively respond to a broad variety of inputs, while only receiving few feedforward and recurrent connections. Few strong, specific recurrent connections then replace many feedforward and less specific recurrent connections, such that the resulting connectivity optimizes the number of required synapses. This suggests that the number of required synapses may be a crucial constraining factor in biological neural networks. Synaptic savings increase with the dimensionality of the encoded variables. We find a trade-off between saving synapses and response speed. The response speed improves by orders of magnitude when utilizing the window of opportunity between excitatory and delayed inhibitory currents that arises if, as found in experiments, spike frequency adaptation is present or strong recurrent excitation is balanced by strong, shortly-lagged inhibition.

Author summary

Neurons represent continuous sensory or intrinsic variables in their joint activity, with rather broad and overlapping individual response profiles. In particular there are often many neurons with highly similar tuning. In the cortex, these neurons predominantly excite each other. We provide a new explanation for this type of recurrent excitation, showing that it can arise in a novel cooperative coding scheme that minimizes the number of required synapses. This suggests the number of required synapses as a crucial constraining factor in biological neural networks. In our cooperative coding scheme, neurons use few strong and specific excitatory connections to share their computations with those neurons that also need it. This way, neurons can generate a large part of their response by leveraging inputs from neurons with similar responses. This allows to replace many feedforward and less specific recurrent connections by few specific recurrent connections. We find a trade-off between saving synapses and response speed. Theoretical estimates and numerical simulations show that specific features of biological single neurons and neural networks can drastically increase the response speed, improving the trade-off.

Introduction

The brain encodes continuous sensory or intrinsic variables in the coordinated activity of populations of neurons. The tuning curves (response profiles) of individual neurons in such populations are rather broad, leading to large overlaps between them [1,2]. Further, there are often many neurons with highly similar tuning. Neuron populations with such features include simple cells in the primary visual cortex (V1) [3,4], head direction cells in the anterior thalamic nucleus [5], tactile neurons in primary somatosensory cortex [6], place cells in the hippocampus [7] and grid cells in the medial entorhinal cortex [8,9]. In machine learning, convolutional networks have overlapping receptive fields (RFs) that tile the input space [10]. RFs similar to those in visual cortex emerge by learning a sparse code for natural images [11], and RFs similar to grid cells emerge through training on navigation tasks [12,13].

Neurobiological data show that neurons with strongly overlapping receptive fields are predominantly excitatorily coupled: Synaptic connections between similarly-tuned excitatory principal neurons are more likely [14], stronger and more often bidirectional [15,16]. In line with this, the strongest incoming synapses provide excitation that matches a neuron's RF [16,17]. Furthermore, highly similarly tuned principal neurons have overall, i.e. including indirect, polysynaptic connections, a net excitatory effect on each other [18,19]. In contrast, if the tuning is barely similar or dissimilar, the net effect is inhibitory.

Such recurrent excitatory connectivity may seem unintuitive from a normative standpoint, as it amplifies noise [20] and can increase response times [21,22]. Previous studies suggested that it may support persistent activity and thus working memory [23,24] and that it may implement complicated priors [25].

Neural networks, however, evolved subject to physiological and physical constraints [26–29], including metabolic cost and available space. Optimizing for specific features can largely determine the neural network and lead to solutions that are in other aspects sub-optimal. A prominent example for this is a recent version of the efficient coding hypothesis [30–34]. It posits that neural networks greedily minimize the number of used spikes or the rate activity, which contribute to metabolic cost. The network connectivity obtained from the optimization is, however, very dense, which is not found in experiments. Further, the coding scheme is "competitive", in the sense that similarly tuned neurons compete for the opportunity to generate spikes. In other words, such neurons take away spikes and activity from each other. This predicts inhibitory couplings between very similarly tuned neurons, contrary to the experimentally observed physiological and effective excitatory interconnectivity between them.

Here, we explore the implications of "cooperative coding" in a neural network. In this newly proposed scheme, neurons avoid replicating computations through feedforward weights whose results are already accessible from the activity of other feature neurons. Instead, each feature neuron performs only a non-redundant feedforward computation. It then achieves the required response by additionally incorporating the results already obtained by similarly tuned feature neurons through recurrent connections. In other words, feature neurons do not independently replicate shared parts of the computations through feedforward weights, but they transmit them through recurrent connections to each other. The resulting connectivity is like-to-like, i.e. strong and effectively excitatory between similarly tuned principle neurons, as observed in experiments. Interestingly the scheme optimizes the number of synapses in a network, while maintaining the required neural network dynamics. Such an optimization differs from the common focus on saving spikes and may be imposed by space restrictions or cost of maintaining synapses [28,35].

Results

To demonstrate the concept of cooperative coding, we consider a layer of feature neurons (output neurons), which receive feedforward input from an input layer as well as recurrent input. The task of the feature neurons is to generate a weighted sum of the inputs with weight strengths that decay exponentially with the distance of an input from the preferred input. We assume that the functionally relevant network response, representing the desired features (outputs), is the steady state activity. The desired outputs are linear functions of the inputs. Neural responses can hence be characterized by linear RFs and implemented by feedforward connectivity alone. Importantly, they can also be implemented using mixtures of feedforward and recurrent input.

We will compare the different network implementations in terms of the space requirement, approximated by the number of required synapses, and in terms of the metabolic cost to keep up the stationary state. Finally we will compare the response times and demonstrate how they can be substantially decreased in networks with spike frequency adaptation (SFA) or balancing inhibition.

Models

In the following, we analyze three concrete examples of cooperative coding: (i) encoding a one-dimensional stimulus, (ii) simultaneously encoding two one-dimensional stimuli with linear mixed selectivity (MS) and (iii) encoding a two-dimensional stimulus. For ease of description, we focus on translationally invariant RFs. (Approximate) translational invariance, meaning that offset RFs have similar shapes, is a common characteristic of experimentally encountered RFs [2,4,5,8,9]. Further, it is a common characteristic of RFs that emerge in machine learning [10,11]. Although the RFs that we consider do not have the precise shape of measured RFs, for example those of simple cells in V1 [36], they share the key properties of localized, overlapping and broadening RFs that tile the represented space. Indeed, RFs of neurons in hierarchically higher layers are often broader and constructed from those in lower layers [37,38].

Encoding a 1D stimulus

As a concrete, analytically tractable model that illustrates how cooperative coding works and can save synapses, we consider RFs that tile the one-dimensional parameter space of a stimulus (see Fig. 1a)). An input neuron j, $j = 1, ..., N$ signals the presence and strength of a stimulus with a specific parameter j by nonzero activity $r_j > 0$. The task of the feature layer is to generate a response that is maximal at the preferred stimulus parameter and then decays exponentially the more different the stimulus becomes from the preferred one. This behavior is qualitatively similar to commonly observed tuning curves such as orientation tuning curves or place fields. We further assume that if multiple stimuli are present, the feature layer responses to their different parameters superpose linearly.

As an example, the inputs may be interpreted as a simple model for the representation of the orientation of a bar in the early visual system. $r_j > 0$ then means that the orientation is within the jth bin of the total orientation range $[0, 180°]$. The transformation from input to features in our model describes the combination of responses from hierarchically lower visual areas to hierarchically higher ones [39]. As another example, the neurons may model the activity of place cells on a periodic, closed track. The transformation then models the transformation from input neurons with smaller place fields to neurons with larger place fields. Such a transformation may take place from the hippocampal dentate gyrus to the downstream area CA3 [40]. In our model, the input generates a

simple encoding of the current location, where input neuron j is active if the animal is in the jth location.

The desired stationary feature layer activity can be expressed as

$$x_i^{\text{resp}} = \sum_{j=1}^{N} \text{RF}_{ij} r_j, \qquad \text{RF}_{ij} = e^{-\frac{|i-j|}{d}} = \gamma^{|i-j|}. \tag{1}$$

Here r_j is the activity of the jth input neuron, $\gamma = \exp(-1/d)$ and d defines the width

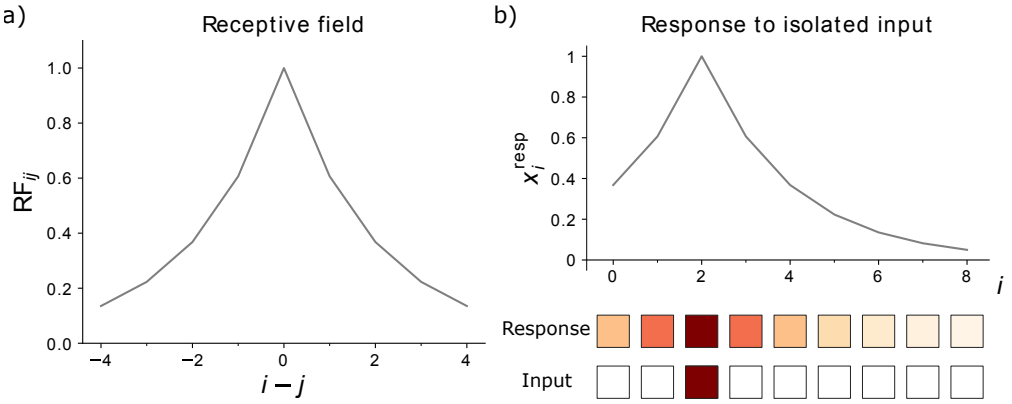

Fig 1: **Receptive field and response of 1D network. a)** The RF of neuron i is peaked at $j = i$ and decays exponentially with $|i - j|$. The receptive field width parameter is $d = 2$. **b)** Top: The network response to an isolated unit input, here located at $j_0 = 2$, has the same shape and amplitude as a neuron's RF. It peaks at $i = j_0$. Bottom: Input and feature neurons are shown with color-coded activities r_j and x_i, respectively. Increasingly dark red color represents higher activity; white squares indicate inactive neurons.

of the RF. We use periodic boundary conditions. For computations with neuron indices, this means that $|i - j|$ means $\min_{n\in\{-1,0,1\}} |i - j + nN|$. There are as many input as feature neurons. We note that, because of the symmetry $\text{RF}_{ij} = \text{RF}_{ji}$, the vector $\text{RF}_{k\cdot}$, describing the RF of feature neuron k, is the same as $\text{RF}_{\cdot k}$, the network response when only input neuron k is active, compare Fig. 1a) and b). Summarized in a formula, we have $x_k^{\text{resp}}|_{r_j=\delta_{jl}} = \text{RF}_{kl} = \text{RF}_{lk} = x_l^{\text{resp}}|_{r_j=\delta_{jk}}$, where k is fixed and l variable.

Feedforward implementation

To model the temporal dynamics of the neurons, we choose a standard simple linear rate network model [41, 42]. The purely feedforward network that generates the response eq. (1) as stationary state is then given by

$$\tau \dot{x}_i(t) = -x_i(t) + \sum_{j=1}^{N} W_{ij}^{\text{ff}} r_j(t), \tag{2}$$

with $W_{ij}^{\text{ff}} = \text{RF}_{ij}$ and a time constant τ. In the stationary state, we have $\dot{x}_i = 0$ for all i and thus

$$x_i = \sum_{j=1}^{N} W_{ij}^{\text{ff}} r_j = \sum_{j=1}^{N} \text{RF}_{ij} r_j = x_i^{\text{resp}}. \tag{3}$$

This state is asymptotically stable and globally attracting; the flow is a contraction to it. These properties follow immediately from the fact that the system is linear and has a

unique fixed point, which is asymptotically stable because all eigenvalues of the matrix specifying the homogeneous differential equation are negative, equal to $-1/\tau$ [43,44]. To approximate the network with a characteristic number of feedforward weights that is smaller than N, we require synapses $W_{ij}^{\mathrm{ff}} = \gamma^{|i-j|}$ only where $|i-j| \leq d$. This defines the RF size $n_{\mathrm{RF}} = 2d+1$ as the number of feedforward synapses per neuron needed to implement the RF within a distance d around its center.

Cooperative implementation

The same stationary neuronal responses can be obtained as the steady state of a recurrent network that uses cooperative coding. It requires only three synapses per feature neuron, two recurrent and one feedforward one. This network's dynamics are given by

$$\tau \dot{x}_i(t) = -x_i(t) + \sum_{j=1}^{N} W_{ij}^{\mathrm{rec}} x_j(t) + \sum_{j=1}^{N} W_{ij}^{\mathrm{ff}} r_j(t) \tag{4}$$

$$= -x_i(t) + w^{\mathrm{rec}}(x_{i+1}(t) + x_{i-1}(t)) + w^{\mathrm{ff}} r_i(t), \tag{5}$$

with weights $w^{\mathrm{rec}} = \frac{1}{\gamma+\gamma^{-1}} = \frac{\gamma}{1+\gamma^2}$ and $w^{\mathrm{ff}} = 1 - 2\gamma w^{\mathrm{rec}} = \frac{1-\gamma^2}{1+\gamma^2}$. If the receptive fields are not narrow (d is not small against 1), the two recurrent connections are strong, in the sense that w^{rec} is not small against 1. Thus the network features strong like-to-like excitation and is driven by feedforward input. One can straightforwardly verify that $x_i = x_i^{\mathrm{resp}}$ is indeed a stationary state of the network, by inserting eq. (1) into eq. (5), see appendix S1. The reason for this is ultimately that the desired response of a neuron i can be largely generated by summing the responses of the two neurons $i \pm 1$ neighboring i, see eq. (9) and Fig. 2b). This is achieved by the recurrent connections. The missing part is contributed by the feedforward input. This state is asymptotically stable as all real parts of the eigenvalues of the matrix defining the homogeneous system are negative, see appendix S1. For broad receptive fields (where $\gamma \lesssim 1$), the recurrent connections are nearly as strong as possible: their sum $2w^{\mathrm{rec}}$ is close to 1, the value beyond which the network becomes unstable. The stationary state is also the only stationary state. Since the system is linear, the state is therefore a global attractor as for the feedforward network [43,44]. Thus, for constant input the network forms this stable response pattern.

Cooperative coding

Cooperative coding can be understood as sharing of the information that an individual neuron obtains from external input specifically with those neurons that also need it. This allows to generate most of the neuronal responses from sparse recurrent connectivity. Especially very similarly tuned neurons will project strongly excitatorily onto each other; oppositely tuned neurons would inhibit each other.

As a concrete example, we introduced the networks eq. (5), where it suffices that each neuron receives input from only one input and two feature neurons. Still, each neuron effectively responds to $\mathcal{O}(d)$ input neurons. This is possible because the feature neurons recurrently share their activity, and hence their access to feedforward input, with their neighbors. These in turn share it with their neighbors, thus propagating it through the network. The network response then forms dynamically through the interplay of feedforward input and recurrent interactions.

The coding relies on the fact that despite very few feedforward and recurrent synapses, poly-synaptic connectivity can still be far-reaching [45,46]: For further clarification of

this point consider the approximate, discretized dynamics [47,48]

$$x_i((n+1)\tau) \approx \sum_{j=1}^{N} W_{ij}^{\mathrm{rec}} x_j(n\tau) + \sum_{j=1}^{N} W_{ij}^{\mathrm{ff}} r_j \tag{6}$$

that lead to the same steady state as the time-continuous dynamics eq. (4). The response to a constant input r after n time constants is

$$x(n\tau) = \left(\mathbb{1} + W^{\mathrm{rec}} + \cdots + (W^{\mathrm{rec}})^{n-1}\right) W^{\mathrm{ff}} r. \tag{7}$$

It is determined by W^{rec} and its higher powers, which reflect the redistribution of feedforward input via poly-synaptic recurrent pathways.

The coding scheme can also be understood as feedforward inputs providing a correction to the response that is mainly constructed from the sparse recurrent input. To clarify this we focus on elementary stationary responses, namely those that are driven by a single unit input from neuron j; the input activity is $r_k = \delta_{kj}$. Responses to more complicated input patterns are weighted linear sums of such elementary responses. Consider feature neuron i and assume that all other neurons already respond correctly. The stationary activity of neuron i in response to a single unit input from neuron j is then RF_{ij}, while those of the other network neurons k is RF_{kj}. Equation (4) with $\dot{x}_i = 0$ implies that RF_{ij} is the sum of the RFs of its presynaptic feature neurons and its feedforward connectivity,

$$\mathrm{RF}_{ij} = \sum_k W_{ik}^{\mathrm{rec}} \mathrm{RF}_{kj} + W_{ij}^{\mathrm{ff}}. \tag{8}$$

For the specific network eq. (5) we have

$$\mathrm{RF}_{ij} = w^{\mathrm{rec}}(\mathrm{RF}_{i-1,j} + \mathrm{RF}_{i+1,j}) + w^{\mathrm{ff}}\delta_{ij}, \tag{9}$$

illustrated in Fig. 2c). In this network, the weighted and summed responses of neuron i's nearest neighbors are thus already very close to neuron i's target response. This is enabled by the specific exponential shape of the RFs. Only for the preferred input of a feature neuron, the response is too low. The neuron corrects for this by recruiting the missing input through a feedforward connection. Such an "explanatory gap" that is left by the recurrent inputs and can be filled by external input is important for cooperative coding, because the output must depend on external input.

Spatial demand and metabolic cost

To compare the efficiency of the introduced implementations, we focus on two cost dimensions: the space needed to implement the network and the metabolic cost of generating the stationary dynamics. As measure for the space needed for the network we take the number of synaptic connections, or, in other words, the L0 norm of the synaptic weight matrix. In the feedforward network eq. (2), it increases linearly with the width d of the RF (if small responses can be neglected). In the recurrent network eq. (5) three synapses per neuron suffice to generate the desired stationary response regardless of the RF size. We show in appendix S1 that the recurrent network eq. (5) therefore minimizes the L0 norm.

For the metabolic cost of generating the stationary dynamics, we can take a multiple of the L1 norm of the synaptic currents (see Discussion). Since in both networks all modeled synaptic currents are excitatory, the L1 norm of synaptic currents equals the total synaptic current. In the stationary state this current is the same in both implementations, because neurons have the same stationary activity; therefore also the cost is the same. We conclude that the metabolic cost for maintaining the stationary network state is the same in both the feedforward and the recurrent network implementation.

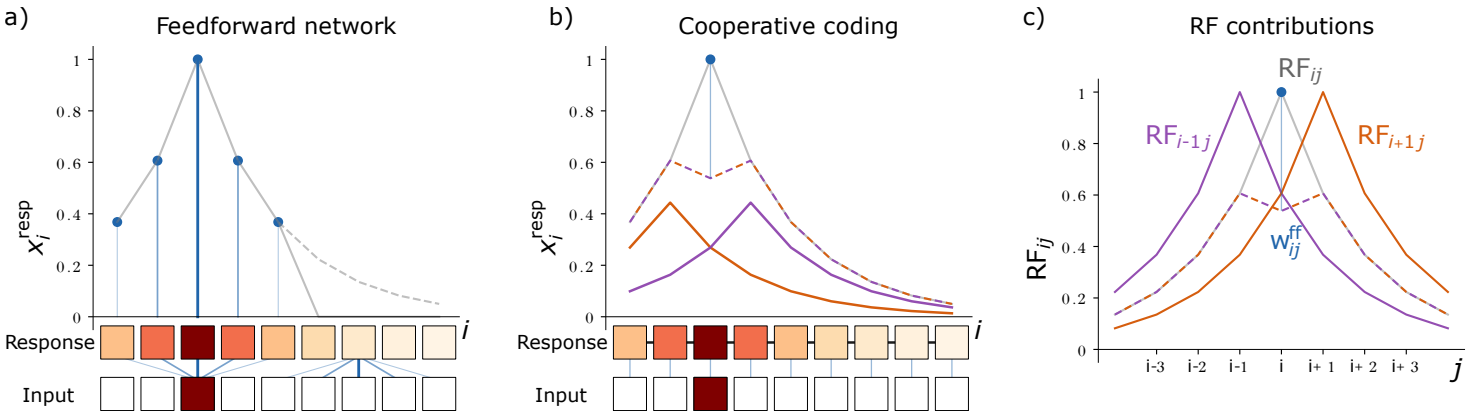

Fig 2: **Schematics of feedforward and cooperatively coding networks.** **a)** Top: In the feedforward network, the response x_i^{resp} (gray solid curve) to an isolated input is fully generated by the neurons' feedforward inputs (blue lines and dots). For the displayed RF width $d = 2$, five neurons receive feedforward input, so that the network response (gray solid curve) represents $\approx 63\%$ of the summed target response (gray dashed curve). Bottom: Feature and input neuron activities as in Fig. 1. Outgoing feedforward synapses from the active input neuron $j = 2$ and incoming feedforward synapses to feature neuron $i = 6$ are shown in blue. **b)** Top: In the cooperatively coding network model, the network response (gray solid curve) is the sum of feedforward input (blue line and dot) and recurrent input (brown-purple dashed curve). For the displayed case of an isolated input, only one neuron receives feedforward input, which induces a part of the stationary response of the most active feature neuron. The rest of the response and all other responses are induced by recurrent input from neighboring neurons. The total recurrent input that each feature neuron receives is the sum of recurrent input from the right (brown solid curve) and left neighbor (purple solid curve). Bottom: Each feature neuron receives one feedforward synapse (blue lines) and two recurrent synapses (black lines, all recurrent connections are bidirectional). **c)** The RF of feature neuron i (RF_{ij} for varying j, gray solid curve) is the weighted sum (brown-purple dashed curve) of the RFs of its left ($\text{RF}_{i-1\,j}$, purple) and right neighbors ($\text{RF}_{i+1\,j}$, brown) plus a contribution from feedforward input (W_{ij}^{ff}, blue line and dot). All shown RFs have width $d = 2$.

Response speed

In the feedforward network, activity converges with the intrinsic time constant τ, which we define as its response time. In the cooperatively coding network, the excitatory recurrent connectivity increases the response time: Fig. 3 shows the dynamics of the response formation. The L1-norm of the deviation of the response from the steady state,

$$L(t) = |x(t) - x^{\text{steady}}|_1, \tag{10}$$

which we use as a loss measure, decays exponentially with time constant

$$\tau_{\text{resp}} \equiv \frac{\tau}{1 - w_{\text{sum}}^{\text{rec}}}, \tag{11}$$

(see Fig. 4), where $w_{\text{sum}}^{\text{rec}} = \sum_j W_{ij}^{\text{rec}}$ is the sum of recurrent weights arriving at (or, equivalently, originating from) a neuron. We define τ_{resp} as the response time of the networks. It scales inversely with the difference of the largest eigenvalue of the network, which is equal to $w_{\text{sum}}^{\text{rec}}$ (cf. appendix S1), from 1. In particular, it depends only on the summed recurrent weights. Equation (11) holds generally, for networks of the type eq. (4) with purely excitatory circulant recurrent weight matrix and convergent dynamics.

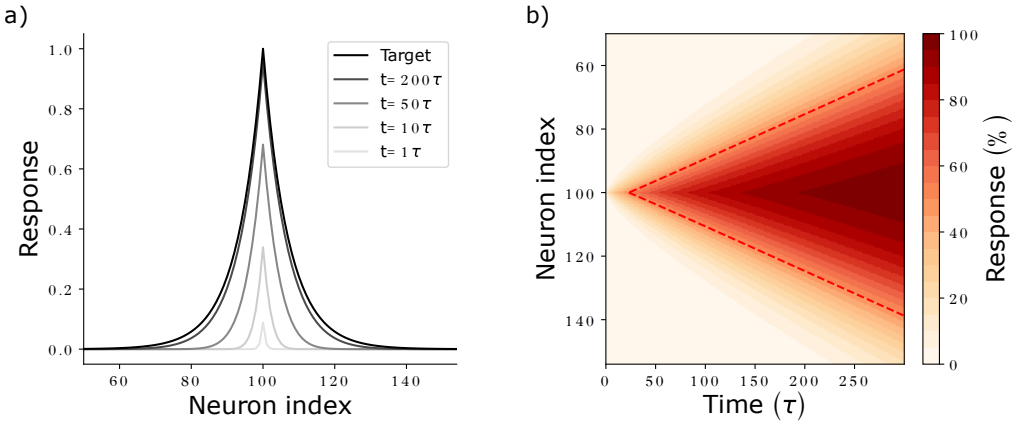

Fig 3: **Response formation and activity propagation. a)** Network activity at different times (shaded curves) after r_{100} has been set from 0 to 1. For long times, network activity approaches the target response (black curve). **b)** Development of the activity of neurons (y axis) with time (x axis), measured relative to their target activities. The diagonal fronts of equal relative activities indicate propagation of activity with constant propagation speed. The time points at which neurons reach 50% of their final activity are connected by a red dashed line. Parameters: $w^{\text{rec}} = 0.5 \cdot 1/(1 - 1/100)$, such that $\tau_{\text{resp}} = 100\tau$ (see eq. (11)), $N = 200$ neurons.

We now specialize the result to networks with nearest-neighbor coupling eq. (5) that generate the RFs eq. (1). Inserting $w^{\text{rec}} = 1/(\gamma + \gamma^{-1})$ and $\gamma = \exp(-1/d)$ relates the response time to the RF width. By approximating $\exp(\pm 1/d) \approx 1 \pm 1/d + (1/2)d^2$ for large d, we obtain $w^{\text{rec}} \approx 1/(2 + 1/d^2)$ and, inserting this into eq. (11),

$$\tau_{\text{resp}} \approx \frac{\tau}{1 - \frac{2}{2 + \frac{1}{d^2}}} = (1 + 2d^2)\ \tau \approx 2d^2\tau \approx \frac{1}{2} n_{\text{RF}}^2 \tau. \tag{12}$$

In the last part of the equation we used that $n_{\text{RF}} = 1 + 2d \approx 2d$ for large d. Equation (12) shows that wide RFs require long equilibration time. This is because they need strong recurrent weights with a largest eigenvalue close to 1. Further the equation reveals the trade-off between response time and number of employed synapses: The feedforward implementation eq. (2) needs n_{RF} synapses and has a response time τ. The recurrent implementation thus saves $n_{\text{RF}} - 3 \approx n_{\text{RF}}$ synapses per feature neuron. Equation (12) shows that the response time increases quadratically in the number of saved synapses, see also Fig. 4b).

The quadratic dependence of the response time on d reflects that, as the RF becomes wider, not only does activity have to spread further, it also spreads more slowly: this is consistent with the idea that the settling of a neuron depends (indirectly) more on activity propagating back from more distant neurons that settle after it.

Faster response with spike frequency adaptation

For activity to rapidly spread through the network, neurons need to be able to cause a large activity change in their neighbors within a short period of time. To achieve this, they need strong recurrent weights. However, recurrent weights are restricted to $w_{\text{sum}}^{\text{rec}} < 1$ to not cause runaway activity. We now show how spike frequency adaptation (SFA) can help ease this conflict and speed up network dynamics. SFA is typical for excitatory principal neurons and induces a reduction of their response to constant inputs in the long run [42,49,50].

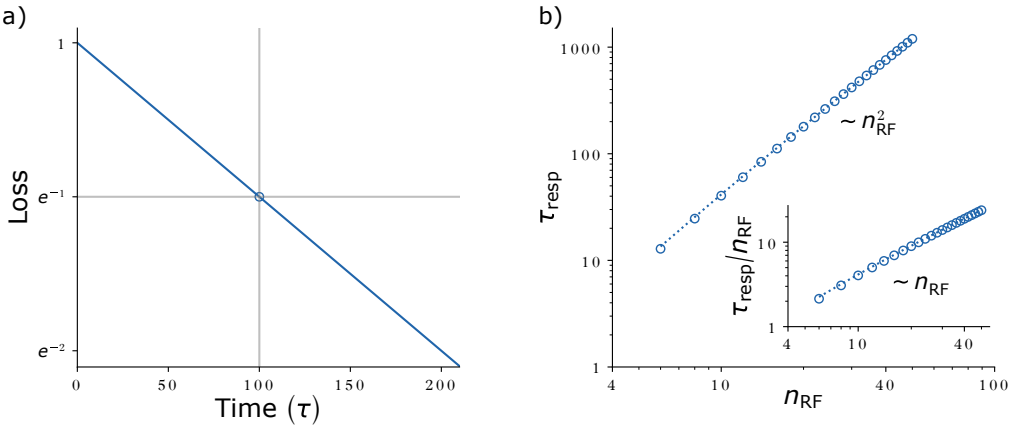

Fig 4: **Loss evolution and response speed - synapse number trade-off. a)** Exemplary loss evolution for a network with $w_{\text{sum}}^{\text{rec,net}} = 0.5 \cdot (1 - 1/100)$ so that $\tau_{\text{resp}} = 100\tau$. Experimentally, τ_{resp} is determined as the time (gray vertical line) at which the loss drops to e^{-1} (gray horizontal line, blue open circle). **b)** Response times τ_{resp} (circles: simulation results; dotted line: analytical solution eq. (12)) for target RFs of different widths n_{RF}. Data was created by scanning n_{RF}, setting $w_{\text{sum}}^{\text{rec,net}}$ to yield a RF of size n_{RF} and determining τ_{resp} from the loss dynamics.

We model SFA through a negative-feedback adaptation current $u(t)$, which is triggered by neuronal activity $x(t)$ and characterized by its scale a_{SFA} and time constant τ_{SFA},

$$\tau \dot{x}_i(t) = -x_i(t) + \sum_{j=1}^{N} W_{ij}^{\text{rec}} x_j(t) + \sum_{j=1}^{N} W_{ij}^{\text{ff}} r_j(t) - a_{\text{SFA}} u_i(t), \tag{13}$$

$$\tau_{\text{SFA}} \dot{u}_i(t) = -u_i(t) + x_i(t). \tag{14}$$

[51–53]. Setting $\dot{x}_i(t) = 0$ and $\dot{u}_i(t) = 0$ yields the steady state. We see immediately that it implies $u_i = x_i$. Inserting this into eq. (13) shows that in the stationary state the spike frequency adaptation results in a stronger leak current, $-(1 + a_{\text{SFA}})x_i$. Dividing by $1 + a_{\text{SFA}}$ yields

$$0 = -x_i + \sum_{j=1}^{N} \frac{W_{ij}^{\text{rec}}}{1 + a_{\text{SFA}}} x_j + \sum_{j=1}^{N} \frac{W_{ij}^{\text{ff}}}{1 + a_{\text{SFA}}} r_j. \tag{15}$$

Consequently, in order to implement the same response as a network without SFA ($a_{\text{SFA}} = 0$, cf. eq. (4)), the recurrent and feedforward weights have to be scaled up by a factor of $1 + a_{\text{SFA}}$. The additional excitatory synaptic input compensates in the steady state the added inhibitory adaptation current.

To understand the network dynamics, it is instructive to consider the limit $\tau_{\text{SFA}} \to 0$ where $u_i(t) \to x_i(t)$ as in the steady state. Inserting this into eq. (13) and again dividing by $1 + a_{\text{SFA}}$ yields an equation equivalent to eq. (4) with shortened neuronal time constant and increased weights,

$$\frac{\tau}{1 + a_{\text{SFA}}} \dot{x}_i(t) \overset{\tau_{\text{SFA}} \to 0}{=} -x_i(t) + \sum_{j=1}^{N} \frac{W_{ij}^{\text{rec}}}{1 + a_{\text{SFA}}} x_j(t) + \sum_{j=1}^{N} \frac{W_{ij}^{\text{ff}}}{1 + a_{\text{SFA}}} r_j(t). \tag{16}$$

We see that a network with arbitrarily fast SFA and appropriately scaled weights has the same dynamics as a network without SFA, but with its time constant reduced by $1 + a_{\text{SFA}}$. This factor only depends on a_{SFA} and is independent of the RF width that

the network implements. We might thus expect that introducing SFA with a given a_{SFA} and small τ_{SFA} causes a constant speedup, but still results in a quadratic dependence of $\tau_{\mathrm{resp}}^{\mathrm{SFA}}$ on n_{RF} (see eq. (12)).

There is, however, an additional possibility: SFA might yield faster dynamics for finite, nonzero τ_{SFA}. This is because then $u_i(t)$ lags behind $x_i(t)$, which creates a temporal 'window of opportunity'. Within this window, the up-scaled weights can mediate strong interactions that are not yet cancelled by the retarded adaptation currents of the receiving neurons. In our networks, this leads to the following concept to exploit SFA: During the initial response phase, strong weights should cause a fast response while SFA keeps the steady state before and after an input change at the desired activity values as well as dynamically stable. In particular, the modified recurrent synaptic weights may then be (and to optimally exploit SFA: should be) so strong that without the SFA current the network dynamics are unstable.

This second possibility applies to our networks: Measuring the response time as a function of τ_{SFA}, we observe that it first decreases when increasing τ_{SFA} from zero and reaches a minimum at a nonzero, optimal value of the SFA time scale (appendix S4). Increasing τ_{SFA} further eventually causes diverging activity, because the retarded adaptation current $u(t)$ becomes so slow that it never compensates the stronger input due to the up-scaled weights. We note that also when keeping τ_{SFA} at a fixed value, there is an optimal nonzero value of the inhibitory feedback strength a_{SFA}. Importantly, we find that introducing SFA with finite τ_{SFA} and optimal a_{SFA} improves the scaling of $\tau_{\mathrm{resp}}^{\mathrm{SFA}}$ with n_{RF} from quadratic as without or with arbitrarily fast SFA to linear.

To incorporate SFA in a cooperatively coding network, we modify the weights in eq. (5) as described above and add the SFA current. For the neuron activities, this yields the dynamical equation

$$\begin{aligned}\tau \dot{x_i}(t) = -x_i(t) &+ (1+a_{\mathrm{SFA}}) w^{\mathrm{rec}} (x_{i+1}(t) + x_{i-1}(t)) \\ &+ (1+a_{\mathrm{SFA}}) w^{\mathrm{ff}} r_i(t) - a_{\mathrm{SFA}} u_i(t);\end{aligned} \tag{17}$$

where the adaptation current obeys eq. (14).

Concerning the use of resources, SFA does not require additional synaptic connections, so the spatial demand of the cooperatively coding network is the same as in the original model eq. (5). The increased weights, however, lead to stronger synaptic currents. Together with the added adaptation currents, this increases the energetic cost of maintaining the stationary state.

Balanced networks

The networks we studied so far had only excitatory synapses, while biological neural networks also have recurrent inhibition, which balances the excitation [32, 54]. These are likely required for a range of reasons, such as ensuring network stability and maintaining irregular spiking activity [55–59]. Given their existence, we here show how inhibition can be used to speed up the network response, in an architecture that still relies on few synapses.

The individual excitatory and inhibitory currents can be much larger than their sum and precisely temporally balanced with a lag smaller than the neuronal time constant [60, 61]. Further, many inhibitory neurons are rather sharply tuned [61–63], sometimes similarly sharply as excitatory ones.

To incorporate inhibition consistent with these experimental findings, we add inhibitory interneurons to the existing network of excitatory feature neurons, which represent principle neurons. The interneurons derive their tuning from the feature neurons via specific recurrent inputs. For concreteness we assume that there are as many

interneurons as feature neurons and that each interneuron follows the activity of one feature neuron with a small delay, τ_{lag}. Equation (4) thus becomes

$$\tau \dot{x}_i(t) = -x_i(t) + \sum_{j=1}^{N} W_{ij}^{\text{rec,E}} x_j(t) + \sum_{j=1}^{N} W_{ij}^{\text{rec,I}} x_j^{\text{I}}(t) + \sum_{j=1}^{N} W_{ij}^{\text{ff}} r_j(t) \tag{18}$$

$$= -x_i(t) + \sum_{j=1}^{N} W_{ij}^{\text{rec,E}} x_j(t) + \sum_{j=1}^{N} W_{ij}^{\text{rec,I}} x_j(t - \tau_{\text{lag}}) + \sum_{j=1}^{N} W_{ij}^{\text{ff}} r_j(t), \tag{19}$$

where $x_i^{\text{I}}(t)$ is the inhibitory activity, which equals the delayed excitatory feature neuron activity $x_i(t - \tau_{\text{lag}})$. $W_{ij}^{\text{rec,I}} \leq 0$ is the coupling from inhibitory neuron j to feature neuron i. We now introduce the state change of feature neuron i between $t - \tau_{\text{lag}}$ and t,

$$\Delta x_i(t) = x_i(t) - x_i(t - \tau_{\text{lag}}), \tag{20}$$

as well as the the sum of recurrent excitation and inhibition

$$W_{ij}^{\text{rec,net}} = W_{ij}^{\text{rec,E}} + W_{ij}^{\text{rec,I}}, \tag{21}$$

which we call net weights. These definitions allow to rewrite eq. (19) as

$$\tau \dot{x}_i(t) = -x_i(t) + \sum_{j=1}^{N} W_{ij}^{\text{rec,net}} x_j(t) - \sum_{j=1}^{N} W_{ij}^{\text{rec,I}} \Delta x_j(t) + \sum_{j=1}^{N} W_{ij}^{\text{ff}} r_j(t). \tag{22}$$

We now insert the values of the cooperatively coding network eq. (5) into the equation and further assume that the interneurons inhibit and balance the same sets of neurons that their driving feature neurons excited, i.e. $W_{ij}^{\text{rec,E}} = w^{\text{rec,E}} (\delta_{i+1,j} + \delta_{i-1,j})$ and $W_{ij}^{\text{rec,I}} = w^{\text{rec,I}} (\delta_{i+1,j} + \delta_{i-1,j})$. Equation (19) then becomes

$$\begin{aligned} \tau \dot{x}_i(t) = -x_i(t) &+ w^{\text{rec,E}} \big(x_{i+1}(t) + x_{i-1}(t)\big) \\ &+ w^{\text{rec,I}} \big(x_{i+1}(t - \tau_{\text{lag}}) + x_{i-1}(t - \tau_{\text{lag}})\big) + w^{\text{ff}} r_i(t) \end{aligned} \tag{23}$$

and eq. (22)

$$\begin{aligned} \tau \dot{x}_i(t) = -x_i(t) &+ w^{\text{rec,net}} \big(x_{i+1}(t) + x_{i-1}(t)\big) \\ &- w^{\text{rec,I}} \big(\Delta x_{i+1}(t) + \Delta x_{i-1}(t)\big) + w^{\text{ff}} r_i(t). \end{aligned} \tag{24}$$

The residual inhibitory interaction, i.e. the inhibitory interaction that is not included in $w^{\text{rec,net}}$, depends on $\Delta x_{i\pm1}(t)$ and therefore only acts when there are activity changes during the preceding brief E-I lag. A total change δx_j in the activity of neuron j causes an integrated postsynaptic activity change in neuron i of $-\frac{\tau_{\text{lag}}}{\tau} W_{ij}^{\text{rec,I}} \delta x_j$ (see appendix S5).

To connect eq. (24) to our previous, unbalanced network eq. (5), we set

$$w^{\text{rec,net}} = w^{\text{rec}}. \tag{25}$$

The dynamical equations then agree if $w^{\text{rec,I}} = 0$ or $\Delta x_i(t) = 0$. The latter is satisfied in the steady state. The steady state is thus independent of the amount of inhibition (given that excitation covaries with inhibition such that $w^{\text{rec,E}} + w^{\text{rec,I}} = w^{\text{rec}}$ as implied by eq. (25)) and it is the same as in eq. (5); its stability can, however, change. We may thus think of $w^{\text{rec,net}}$ as defining the RF, and of $w^{\text{rec,I}}$ as affecting the dynamics. During the build-up of the response strong excitation ramps up slightly before the balancing inhibition. The analytical solution of eq. (24) and our stability analysis (see next section and Fig. 9) show that throughout this window of opportunity excitation may be up to approximately τ/τ_{lag} times larger than the net interaction without destabilizing the network. This strong interaction allows a much quicker propagation of activity, convergence to the steady state and decay of the loss function.

Spatial demand and metabolic cost in the balanced network

Compared to the unbalanced network, the balanced network requires three additional synapses per principle feature neuron, one E-to-I and two I-to-E synapses, i.e. a total of six synapses. This is again independent of the RF width, such that for large RFs, the balanced, cooperatively coding network still saves synapses compared to the feedforward network. It requires additional space for the inhibitory neurons, which may, however, be needed for other purposes anyways.

Also the metabolic maintenance cost increases, since there are more neurons. Further, there is an increased metabolic cost to sustain the synaptic currents in the stationary state: In this state, large parts of the excitatory and inhibitory currents cancel to give rise to a net current that equals the one in the purely excitatory recurrent network, see eq. (21) and eq. (25). In the L1 norm of the synaptic currents, the excitatory currents and the absolute inhibitory currents, however, add up. The metabolic cost thus increases by the amount of excitatory and inhibitory currents that cancel each other.

Response speed in the balanced network

Under some additional assumptions, the evolution of the L1-loss eq. (10) can be analytically approximated as the solution of a linear delay differential equation, see appendix S6 for details. The resulting dynamics are those of a damped oscillator, see Fig. 5a): For weak inhibition, they are 'overdamped' in the sense that they are well described by the sum of two exponentials with different decay rates. At a specific intermediate inhibitory strength, the two decay rates agree and we have 'critical damping'. For stronger inhibition the dynamics are 'underdamped' in the sense that the loss behaves as the absolute value of an oscillation with exponentially decaying amplitude. Overly strong inhibition causes divergence of the dynamics.

In the overdamped regime, the smaller decay rate is the relevant one, as it dominates the speed of the decay for longer times. The larger decay rate rather describes how quickly faster dynamics, that may be present due to the initial conditions, are suppressed and the dynamics converge to the slower mode. The smaller decay rate increases when the strength of inhibition approaches its critical value. The same holds for the single decay rate in the oscillatory regime. At the critical inhibitory strength the overall decay of the loss is thus fastest, see Fig. 5b). We find analytically that its decay time constant is approximately proportional to the geometric mean of the response time in absence of inhibition and the inhibitory delay,

$$\tau_{\text{resp}}^{\text{bal},c} \approx \sqrt{\frac{\tau_{\text{resp}}\tau_{\text{lag}}}{2}}, \tag{26}$$

(cf. eq. (89) in appendix S6); the superscript "c" indicates that the result holds for critical inhibitory strength.

Importantly, this implies that the scaling of the response time with the receptive field width and size improves compared to the purely excitatory network. This is because $\tau_{\text{resp}}^{\text{bal},c} \sim \sqrt{\tau_{\text{resp}}}$. Inserting eq. (12) into eq. (26) for the 1D network, we obtain

$$\tau_{\text{resp}}^{\text{bal},c} \approx \sqrt{\frac{(1+2d^2)\ \tau\tau_{\text{lag}}}{2}} \approx d\sqrt{\tau\tau_{\text{lag}}} \approx \frac{1}{2} n_{\text{RF}} \sqrt{\tau\tau_{\text{lag}}}, \tag{27}$$

which is only linear in the RF width d and size n_{RF}, instead of quadratic as in the case of $w_{\text{sum}}^{\text{rec,I}} = 0$, compare eq. (27) with eq. (12) and in Fig. 6a) the red and blue dotted curves. As a consequence also the speedup gained through the inhibition, $\tau_{\text{resp}}/\tau_{\text{resp}}^{\text{bal},c}$, increases for wider RFs. We finally note that the interactions mediated by $W^{\text{rec,I}}$ can also be thought of as implementing an excitatory transmission of activity changes: an

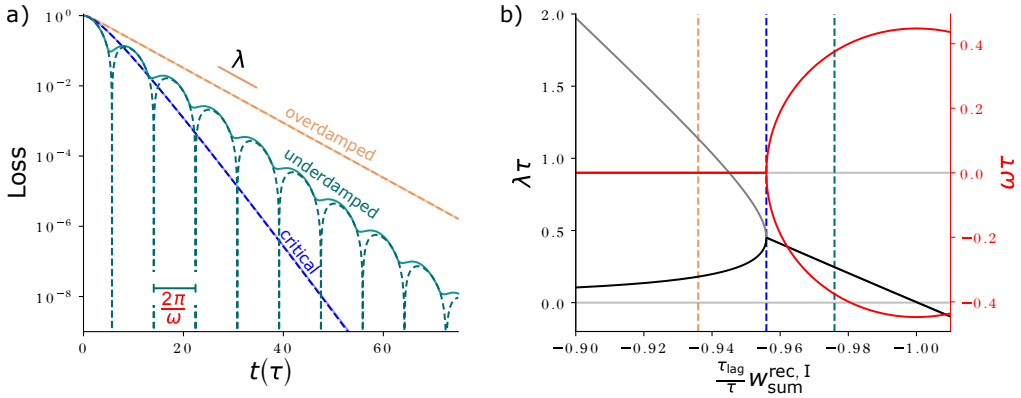

Fig 5: **Loss evolution for different inhibitory strengths. a)** Loss evolution (dashed: analytical approximation (cf. appendix S6, eqs. (65) and (75), partly occluded; solid: network simulation) for inhibitory strengths that are slightly weaker (orange), equal (blue) or slightly stronger (teal) than the critical strength, on a logarithmic scale. The slope of the decay is given by λ (see b), explicitly highlighted for the overdamped dynamics. The oscillation period of the underdamped dynamics is $T_{\text{osci}} = 2\pi/\omega$. In case of oscillations, the analytic approximation briefly reaches zero loss once in a period (sharp dips in dashed curve). In the network simulation there is also a pronounced oscillation, but there always remains a finite error (solid, see appendix S6). **b)** Real part (decay rate λ, black/gray) and imaginary part (oscillation frequency ω times ± 1, red) of the complex frequency of the exponential loss evolution, scaled by τ. For weak inhibition there are two exponentially decaying modes (λ, black and gray curve). At the critical inhibitory strength (blue dashed vertical line) there is only a single decay rate and no oscillation. The decay rate (in the overdamped case: of the relevant slower-decaying mode) is maximized. For stronger inhibition, network activity begins to oscillate (nonzero ω, red), and diverges for $\frac{\tau_{\text{lag}}}{\tau} w_{\text{sum}}^{\text{rec,I}} < -1$, where λ becomes negative. Dashed vertical lines show the inhibitory strengths scaled by τ_{lag}/τ for the curves in a) ($(\tau_{\text{lag}}/\tau) w_{\text{sum},c}^{\text{rec,I}} + 0.02$, $(\tau_{\text{lag}}/\tau) w_{\text{sum},c}^{\text{rec,I}}$, $(\tau_{\text{lag}}/\tau) w_{\text{sum},c}^{\text{rec,I}} - 0.02$). Parameters: $w_{\text{sum}}^{\text{rec,net}} = 0.99$, $\tau = 1$, $\tau_{\text{lag}} = 0.1$, and $N = 200$ for the network simulation.

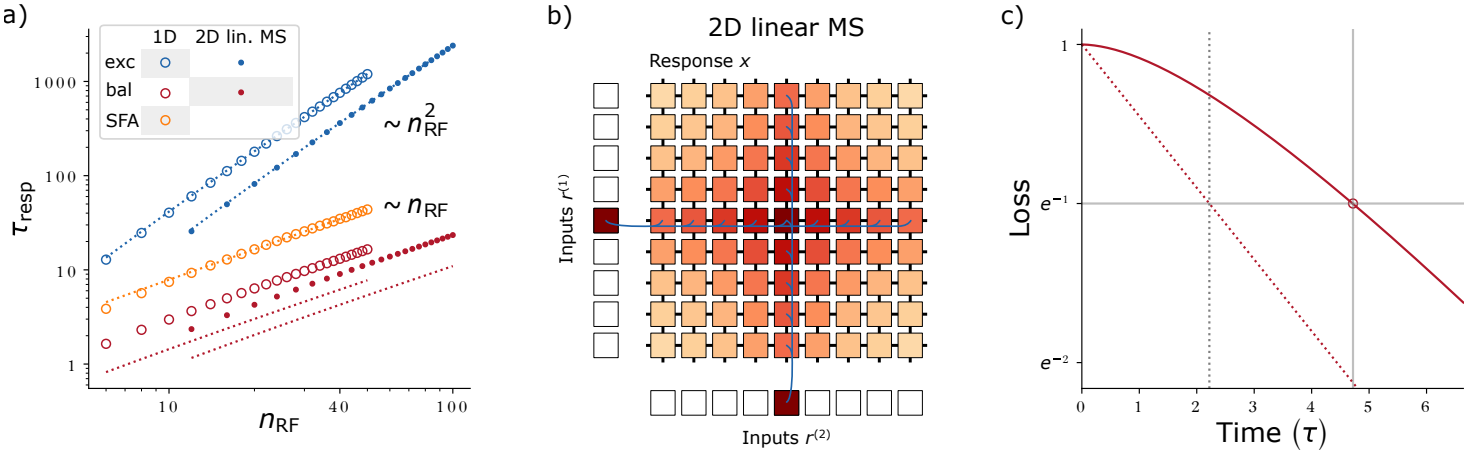

Fig 6: **Response speed of networks with inhibition and linear MS. a)** Response times for the 1D network and for the 2D linear MS network. The quadratic scaling of τ_{resp} with n_{RF} for the excitatory networks (blue) can be improved to a linear dependence by introducing balancing, delayed inhibition (red) or SFA (orange). Open (1D network) and filled (MS network) circles display numerical results. Alike colored dotted (1D network) or continuous (MS network) curves show theoretical estimates (eqs. (12), (27), (33) and (34)) or, for the SFA network, fit results (monomial fit: $\tau_{\text{resp}}^{\text{SFA}}(n_{\text{RF}}) = 0.66724(n_{\text{RF}})^{1.07063}$). We use the slowest decaying eigenmode to theoretically estimate the response times (see c). Since the balanced networks are not initialized in this eigenmode (in contrast to the purely excitatory networks), the numerically measured response times (red markers) lie above the theoretical values (red lines). **b)** Schematic of a 2D network with linear MS. Feature neurons are arranged on a two-dimensional grid (labeled 'Response x'). Each receives feedforward input from two arrays of input neurons (labeled 'Inputs $r^{(1|2)}$') and four recurrent inputs. Feedforward and recurrent synapses are shown in blue (exemplarily) and black, respectively. Input and feature neuron activities are color-coded. The (linear) network response is the sum of the responses to input one and input two. **c)** Exemplary loss evolution for a 1D network with lagged inhibition. Due to the temporally constant initialization ($x_i(0) = 0$, $\Delta x_i(0) = 0$), the network activity (solid red curve) converges initially more slowly than the network's slowest eigenmode (dotted red line). The experimentally measured response time (continuous vertical gray line) is defined as the time when the loss has decayed by $1/e$ (red open circle, horizontal gray line, see also Fig. 4a). It is larger than that of the network's eigenmode (dotted gray line), which we use as analytical estimate of the response time. We created the data in a) by scanning n_{RF}, setting $w_{\text{sum}}^{\text{rec,net}}$ to yield a RF of size n_{RF}, setting $w_{\text{sum}}^{\text{rec,I}}$ to 0 or its critical value, and determining τ_{resp} or $\tau_{\text{resp}}^{\text{bal}}$ from the loss dynamics. For the SFA network we set $\tau_{\text{SFA}} = \tau$, scanned a_{SFA} and used the value that minimized the temporally integrated loss.

activity change in neuron j adds an activity change with the same sign to neuron i, because $-W^{\mathrm{rec,I}}$ in eq. (22) is positive. Thus activity changes in different neurons in the network amplify each other. In the limit of small τ_{lag}, the temporal derivative is transmitted, see appendix S5.

Linear mixed selectivity

Neurons often respond selectively to more than one stimulus or input feature [64]. This phenomenon is called mixed selectivity (MS). Here we consider linear MS [64,65], where neuronal responses are linear functions of multiple stimuli. Concretely, neurons with activities x_{ij}, $i,j = 1,\cdots,N$ are arranged on a two-dimensional grid and respond with equal selectivity to two input features, represented by input neurons $r_k^{(1)}$ and $r_l^{(2)}$ with $k,l = 1,\cdots,N$,

$$x_{ij}^{\mathrm{resp}} = \sum_{k=1}^{N} \mathrm{RF}_{ijk}^{(1)} r_k^{(1)} + \sum_{l=1}^{N} \mathrm{RF}_{ijl}^{(2)} r_l^{(2)}. \tag{28}$$

Due to the linearity in the input representation and in the network, the total response is the sum of the responses to the single input features. We take the grid axes to be aligned with the stimulus dimensions, so that the first index in x_{ij} determines its response to $r^{(1)}$ and the second that to $r^{(2)}$. We model this dependence as the same localized, exponentially decaying shape as for the 1D network (cf. Fig. 6b)),

$$\mathrm{RF}_{ijk}^{(1)} = \gamma^{|i-k|} \qquad\qquad \mathrm{RF}_{ijk}^{(2)} = \gamma^{|j-k|}. \tag{29}$$

The desired network response can be generated as the steady state of a recurrent network that is equivalent to the 1D network eq. (5) in each dimension of the 2D grid (see next section),

$$\begin{aligned}\tau \dot{x}_{ij} = &-x_{ij} + w^{\mathrm{rec,MS}}(x_{i+1,j} + x_{i-1,j} + x_{i,j+1} + x_{i,j-1}) \\ &+ w^{\mathrm{ff,MS}}(r_i^{(1)} + r_j^{(2)}),\end{aligned} \tag{30}$$

with the modified constants $w^{\mathrm{rec,MS}} = \frac{w^{\mathrm{rec}}}{1+2w^{\mathrm{rec}}} = \frac{\gamma}{(1+\gamma)^2}$ and $w^{\mathrm{ff,MS}} = \frac{w^{\mathrm{ff}}}{1+2w^{\mathrm{rec}}} = \frac{1-\gamma^2}{1+\gamma^2}$. Each neuron receives two external inputs and is connected to its nearest neighbors along each stimulus axis. The network has thus only six synapses per neuron, regardless of the RF width. Also a feedforward network where each dimension of the 2D grid is equal to the 1D network eq. (2) generates the desired response. This implementation requires $n_{\mathrm{RF}}^{\mathrm{MS}} = 2n_{\mathrm{RF}} = 2(2d+1)$ synapses per neuron, a number that increases linearly with the RF width.

Mapping to a 1D system

In the following, we trace the network dynamics eq. (30) back to those of the 1D system eq. (5). Due to the linearity of eq. (30), network responses again superpose. It thus suffices to study the network in the case where only one input neuron is active: we choose $r_i^{(1)}$, which specifies a property of the first stimulus, to be nonzero. Since the input is independent of j, the dynamics eq. (30) are (for initial conditions homogeneous in j such as $x_{ij}(0) = 0$) independent of j, $x_{ij}(t) = x_i(t)$. The recurrent inputs $w^{\mathrm{rec,MS}}(x_{i,j+1}(t) + x_{i,j-1}(t)) = 2w^{\mathrm{rec,MS}}x_i(t)$ then simply amount to a modification of the leak current to $-(1-2w^{\mathrm{rec,MS}})x_i$,

$$\tau \dot{x}_i = -(1-2w^{\mathrm{rec,MS}})x_i + w^{\mathrm{ff,MS}} r_i^{(1)} + w^{\mathrm{rec,MS}}(x_{i+1} + x_{i-1}) \tag{31}$$

After dividing by $(1-2w^{\mathrm{rec,MS}})$, the differential equation for x_i becomes

$$a\tau\dot{x}_i = -x_i + aw^{\mathrm{rec,MS}}(x_{i+1}+x_{i-1}) + aw^{\mathrm{ff,MS}}r_i^{(1)}, \tag{32}$$

where we introduced $a=(1-2w^{\mathrm{rec,MS}})^{-1}$ for brevity. With the values of the constants $w^{\mathrm{rec,MS}}$ and $w^{\mathrm{ff,MS}}$ highlighted after eq. (30) this is equivalent to the one-dimensional network dynamics eq. (5) up to a different neuronal time constant $a\tau$ instead of τ. (We note that we obtained the modified constants such that this holds. For example, equating the prefactors of the recurrent term in eq. (32) and eq. (5) gives $w^{\mathrm{rec}} = aw^{\mathrm{rec,MS}} = (1-2w^{\mathrm{rec,MS}})^{-1}w^{\mathrm{rec,MS}}$, which then can be solved for $w^{\mathrm{rec,MS}}$.) As a direct consequence, while the 1D network must have recurrent coupling strength of $w^{\mathrm{rec}} < 0.5$ for being stable, the 2D MS network must have $w^{\mathrm{rec,MS}} < 0.25$. This is because in the MS network, a neuron receives direct recurrent input from four nearest neighbors instead of two as in the 1D case.

Equation (32) means that the RF of the MS network, along one axis, has the same shape and width d as the equivalent one-dimensional network. In particular, d is related to the recurrent weight strength $aw^{\mathrm{rec,MS}}$ via $aw^{\mathrm{rec,MS}} = w^{\mathrm{rec}} = \frac{\gamma}{(1+\gamma)^2}$ and $\gamma = \exp(-1/d)$; the two RF components in eq. (29) are the same as the RFs in eq. (1), for example $\mathrm{RF}^{(1)}_{ijk} = \mathrm{RF}_{ik}$.

Response speed

From the mapping of the MS to the 1D system, eq. (32), we see that the MS dynamics behaves in response to a single input like the 1D dynamics with the neuronal time constant τ enlarged by a factor of a. The response time is thus given by eq. (12), but with enlarged neuronal time constant, $\tau \to a\tau$. For sufficiently large d, we have $\gamma \approx 1$ (reflecting the spatially slow RF decay), $w^{\mathrm{rec}} \approx 1/2$, $w^{\mathrm{rec,MS}} \approx 1/4$ and thus $a \approx 2$. The scaling of the response time with the RF width d and size $n^{\mathrm{MS}}_{\mathrm{RF}}$ is thus again quadratic,

$$\tau^{\mathrm{MS}}_{\mathrm{resp}} \approx (1+2d^2)\,2\tau \approx 4d^2\tau \approx \frac{1}{4}\left(n^{\mathrm{MS}}_{\mathrm{RF}}\right)^2\tau. \tag{33}$$

In the last equation we used $n^{\mathrm{MS}}_{\mathrm{RF}} = 2(2d+1) \approx 4d$. Compared to the 1D case (eq. (12)), the response time as a function of d is therefore larger by a factor $a \approx 2$. In contrast, it is smaller by a factor $1/2$ as a function of the RF size, compare eq. (33) with eq. (12) and the blue continuous and dotted curves in Fig. 6a). In other words: the trade-off between response time and number of needed synapses improves for sufficiently large RFs by a constant factor of about $1/2$ compared to the 1D network. This is because the MS network effectively implements two 1D RFs (eq. (28)).

Balanced network

We now incorporate the effect of inhibitory neurons into the MS network. As in the 1D case, we assume that the generated inhibition precisely tracks excitation with a short time delay. We thus add to each recurrent excitatory connection an inhibitory one that is slightly delayed. This results in a delayed differential equation like eq. (24) for the balanced MS network dynamics. The parameters are given by those of the 1D balanced system up to a factor $a = 1 + w^{\mathrm{rec,net}}_{\mathrm{sum}}$, like in eq. (32). Further, it is again sufficient to study the response dynamics to a single input, which can be reduced to those of the 1D balanced network eq. (24) with adapted parameters. As in the purely excitatory case, for a fair comparison of response times, we consider MS and 1D networks with the same neuronal time constant τ. The effective time constant of the MS dynamics are then $a\tau$. Therefore the response time of the MS network are given by those of the 1D network

eq. (27) with neuronal time constant τ replaced by $a\tau \approx 2\tau$,

$$\tau_{\text{resp}}^{\text{bal},c,\text{MS}} \approx \sqrt{(1+2d^2)\ \tau\tau_{\text{lag}}} \approx \sqrt{2}d\sqrt{\tau\tau_{\text{lag}}} \approx \frac{1}{2\sqrt{2}} n_{\text{RF}}^{\text{MS}} \sqrt{\tau\tau_{\text{lag}}}. \tag{34}$$

At the critical inhibitory strength, the response time thus scales again with the square root of the response time of the network without inhibition (eq. (26)). Therefore it scales linearly with the receptive field width d and size $n_{\text{RF}}^{\text{MS}}$. The response time is as a function of the RF size by a factor of about $1/\sqrt{2}$ smaller than that in the 1D case, eq. (27), see Fig. 6a), red continuous and dotted curves. In other words, the trade-off between response time and number of required synapses improves by a factor of $1/\sqrt{2}$. This is again because the MS network effectively implements two RFs in the MS case; the RF size doubles for the same width compared to the 1D network.

As for the 1D stimulus, the balanced networks have twice as many neurons and additional synapses: Each (excitatory) feature neuron drives one inhibitory neuron, which mirrors its activity. This inhibitory neuron in turn forms inhibitory synapses to the four nearest neighbors that its presynaptic feature neuron excites. In total, the balanced, cooperatively coding network thus requires eleven synapses per feature neuron, instead of six for the unbalanced network. This is independent of the RF width, such that the balanced, cooperatively coding network save synapses for sufficiently wide RFs.

Higher-dimensional linear MS

We can straightforwardly extend the introduced scheme to networks that have MS with $P > 2$ stimuli. Neurons are then arranged on a hyper-grid with one grid axis per stimulus dimension, so that N^P feature neurons respond to PN input neurons. In the cooperatively coding network, each neuron receives P feedforward and $2P$ recurrent inputs, requiring a total of $3P$ synapses per neuron. The feedforward network, in contrast, needs for each stimulus dimension $2d+1$ synapses, in total $P(2d+1)$ synapses per neuron. The number of saved synapses thus grows linearly with the number of encoded stimulus dimensions and the receptive field width.

Encoding a 2D stimulus

We finally consider the encoding of a two-dimensional stimulus, with both input and feature neurons arranged on a two-dimensional grid, see Fig. 7a). Two-dimensional input appears for example in vision [36] or planar navigation tasks [66]. Each feature neuron responds to inputs that are close to its preferred input in both stimulus dimensions. The RFs of neighboring neurons thus overlap and neuronal responses tile the represented stimulus space. A purely excitatory cooperative coding network generating such activity as stationary state is given by

$$\tau\dot{x}_{ij} = -x_{ij} + w^{\text{rec,2D}}(x_{i+1,j} + x_{i-1,j} + x_{i,j+1} + x_{i,j-1}) + w^{\text{ff,2D}} r_{ij}. \tag{35}$$

It has the same recurrent connectivity as the network with linear MS eq. (30), but the feedforward input is arranged on a grid. The activity of feature neuron ij in the stationary state is

$$x_{ij}^{\text{steady}} = \sum_{kl=1}^{N} \text{RF}_{ijkl} r_{kl}. \tag{36}$$

The neuron thus responds to a combination of two input features represented by input neurons r_{ij} with $i,j = 1,\cdots,N$. The RF is explicitly given by

$$\text{RF}_{ijkl} \approx c \cdot K_0(\gamma^{2\text{D}} \rho_{ijkl}), \tag{37}$$

with $c = \frac{1}{2\pi}\frac{w^{\mathrm{ff,2D}}}{w^{\mathrm{rec,2D}}}$, $\gamma^{\mathrm{2D}} = \sqrt{\frac{1-4w^{\mathrm{rec,2D}}}{w^{\mathrm{rec,2D}}}}$ and $\rho_{ijkl} = \sqrt{|i-k|^2 + |j-l|^2}$. K_0 is the zeroth modified Bessel function of the second kind, which decays with distance ρ approximately as $K_0(\rho) \approx \frac{\pi}{2}\mathrm{e}^{-\rho}/\sqrt{\rho + 1/8}$ [67]. The RF is thus approximately radially symmetric. The RF size depends on $w^{\mathrm{rec,2D}}$ and the response amplitude also on $w^{\mathrm{ff,2D}}$. The network requires 5 synapses per neuron.

We also constructed a balanced network by introducing for each excitatory recurrent input a delayed inhibitory one, like in the balanced 1D and MS networks. A balanced implementation with explicit inhibitory interneurons requires twice as many neurons and synapses than the purely excitatory network: it requires additionally one interneuron per principle neuron, one inhibitory synapse for each excitatory recurrent synapse and one synapse from each principle neuron to its corresponding interneuron.

Response speed

To estimate the dependence of the response speed on the RF size, we first need to appropriately adapt the definition of the RF size, which we introduced after eq. (3). For the one-dimensional network, this definition can be reformulated as follows: we count the number of synapses that are necessary to generate the largest (around the center) RF responses such that these responses summed together amount to a fraction of about $1 - e^{-1} \approx 63\%$ of the summed non-truncated RF. Accordingly, for the 2D network at hand we define the RF size as the number of feedforward synapses that are necessary to implement the largest RF entries, such that together they account for a fraction of approximately 63% of the summed nontruncated RF. We denote the so-defined receptive field sizes by $\hat{n}_{\mathrm{RF}}$.

As for the 1D and linear MS networks, the response time with or without lagged inhibition depends only on the summed excitatory weights or on the summed net and inhibitory weights. It is thus given by eq. (11) or by eq. (26) in terms of $w^{\mathrm{rec}}_{\mathrm{sum}} = 4w^{\mathrm{rec,2D}}$ or in terms of the (alike obtained) $w^{\mathrm{rec,net}}_{\mathrm{sum}}$ and $w^{\mathrm{rec,I}}_{\mathrm{sum}}$. Fig. 7b) shows that the scaling of the response time with the RF size is linear for unbalanced and square-root-like for balanced networks. We give a geometric argument for this general scaling in the Discussion (subsection 'Multi-dimensional stimuli'). The scaling is more economical than for the 1D and 2D linear MS networks, cf. Fig. 6.

Discussion

In this work, we have studied networks that encode continuous variables with neurons that have overlapping response properties. We developed a cooperative coding scheme that enables them to share and distribute computations among similarly-tuned neurons, crucially using (net) excitatory connections. For the simplest considered networks this sharing minimizes the number of required synapses while the total amount of synaptic current remains the same as in a purely feedforward implementation. For networks of neurons that represent higher-dimensional stimuli, the number of saved synapses is especially large. The saving of synapses comes at the cost of longer response times. We find, however, that neurons with spike frequency adaptation and neurons in networks in which excitation is largely balanced by delayed inhibition can use the window of opportunity between the arrival of excitation and inhibition to significantly speed up their convergence to the steady state response, decreasing response times by orders of magnitude and improving their scaling with RF size.

RF shape and more complex networks The exponentially decaying RFs that we consider act as tractable models for experimentally encountered localized, overlapping and broadening RFs [2,38,40]. They allow to elegantly illustrate how neurons can use

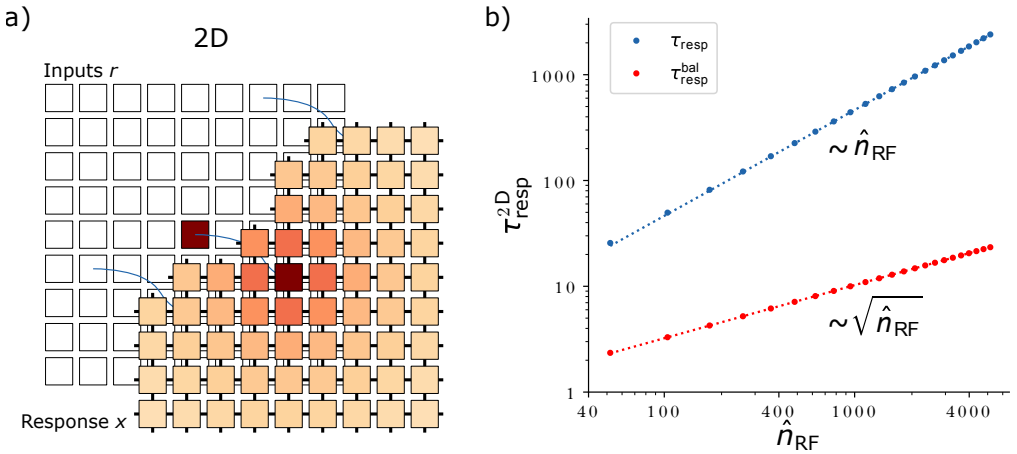

Fig 7: **2D network schematic and response times versus RF size. a)** Schematic of a two-dimensional network responding to a two-dimensional stimulus. Feature neurons (labeled 'Response x') and input neurons (labeled 'Inputs r') are arranged on two-dimensional grids. In the cooperatively coding network each feature neuron receives one feedforward and four recurrent inputs; activities and shown connections are color-coded as in Fig. 2. **b)** Response times in the 2D network increase linearly (without inhibition, blue. Monomial fit: $\tau^{2D}_{resp} = 0.46115\ (\hat{n}_{RF})^{1.00085}$) or square-root-like (with inhibition, red. Monomial fit: $\tau^{bal,2D}_{resp} = 0.31995\ (\hat{n}_{RF})^{0.50202}$) with the RF size $\hat{n}_{RF}$. Dotted lines represent the monomial fits. Data was created by scanning $w^{rec,net}_{sum}$, setting $w^{rec,I}_{sum}$ to 0 or its critical value, and determining $\hat{n}_{RF}$ and τ_{resp} or τ^{bal}_{resp}, respectively, from the response curves after network activity converged.

recurrent interactions to cooperatively share feedforward information and shape the network response. However, experimentally measured RFs have different and often more complex shapes [7–9,36]. We are optimistic that these can still be approximated by the steady state of a cooperatively coding recurrent network with sparse connectivity, although more synapses will be required. Determining the necessary network parameters might involve minimizing a loss with L0 regularization, which is challenging.

Related work Conceptually, our 1D model is a ring model. Models of this type have been proposed to model orientation selectivity in the visual cortex [23,52] (see also [68]), head direction cells [69] and spatial memory [70]. Similarly, our 2D and 2D mixed selectivity models have a toroidal or, when removing the periodicity, a planar structure. Such networks may be important for spatial navigation [66]. With appropriate coupling, ring-like networks can have two different dynamical regimes [42]: (i) an input-driven regime where there is a single, homogeneous ground state, which is assumed in absence of input and (ii) a regime of bump attractors, where there is spatially localized, persistent activity in absence of input. Our networks are linear, therefore without input there cannot be multiple stationary activity patterns whose amplitudes are asymptotically stable. This is because each multiple of a stationary solution is a solution as well. We thus work in the input-driven regime, with a single stable zero ground state. Previous models have broad coupling fields or ranges of coupling probabilities, equivalent to many recurrent synaptic connections that extend over neurons with quite different preferred stimuli [23,42,52,66,69,70]. In contrast, in cooperative coding networks, we have very sparse synaptic connections between neurons with highly similar tuning.

In this work we considered the encoding of continuous variables in a scheme that minimizes the total number of required synapses. Relatedly, ref. [71] investigates the

problem of classifying discrete patterns in large networks of neurons with limited and fixed in-degrees. It solves the problem by introducing an intermediate layer consisting of interconnected perceptrons, which each receive only part of the overall input. This intermediate layer may be seen as analogous to our feature layer. In the mainly studied binary classification task, the intermediate layer is equipped with excitatory connections. The assumed connectivity is untuned except for being overcritically excitatory such that it drives neurons into saturation. In the stationary response to a pattern then all (more precisely: nearly all [72]) neurons in the network assume the same output, either positive or negative one. This allows a readout neuron to perform binary classification with sparse readout weights. A refined architecture combines several such intermediate layers in parallel. This yields an intermediate layer with several subpopulations, which code in binary manner, allowing for classification with a binary vector. Another way highlighted to achieve this is by using as intermediate layer a Hopfield network with recurrent connections that are randomly realized with fixed probability. Since the binary coding network and the network with several subpopulations have specific like-to-like connectivity to save feedforward connections, they realize cooperative coding in our sense, in a binary manner. By choosing the recurrent coupling probability of the Hopfield network intermediate layer such that stronger positive and negative weights (which connect similarly and oppositely tuned neurons, respectively) are realized with larger probability, one could implement cooperative coding in them as well. Ref. [73] find that local recurrent connectivity in Hebbian assemblies of spiking neurons can reduce the number of feedforward connections between assemblies required for memory replay. The total number of synapses in their model is, however, minimized by a purely feedforward architecture. They argue that one benefit of sparse feedforward connectivity, augmented through local recurrence, might be enabling one-shot learning.

The signature of cooperative coding in the networks that we investigated and in more general ones is that the network trades feedforward and less specific recurrent synapses for recurrent synapses mediating on average net-excitatory recurrent interactions between similarly tuned neurons (and net-inhibitory interactions between highly anti-tuned neurons). Consistent with this, intermediate-depth ML networks featuring recurrent and feedback connections can match the performance of much deeper feedforward networks while requiring less units and parameters [74]. It would be interesting to investigate whether the recurrent connectivity in such networks is also like-to-like. If yes, this would indicate that cooperative coding naturally appears also in ML networks. It may be helpful in particular in convolutional networks to save feedforward connections and rely on very sparse recurrent connectivity instead.

Response speed in excitatory 1D networks In our most simple, purely excitatory cooperatively coding networks the response is slowed down compared to that of single neurons due to recurrent excitation, which implements a positive feedback loop. This feedback loop increases the eigenvalues of the matrix governing the differential equation of the dynamics and increases the duration of the response to single pulses. The prolonged response to a single short pulse adds up for prolonged inputs and leads to their amplification as well. This type of amplification has been termed "Hebbian amplification" in [21].

Networks with SFA To speed up the response, we first introduced SFA, a typical feature of excitatory principal neurons [42, 49, 50]. Our model is a slightly simplified version of that in [51] and the same as in [52, 53].

In the networks with SFA excitation still dominates, i.e. we have on the one hand Hebbian amplification as well. On the other hand, we have "balanced amplification" [21]: Each neuron is a 2-dimensional dynamical system, the network consists of N of them;

they are excitatorily coupled. Without the inhibitory feedback from the adaptation the dynamics of the purely excitatory neuron population would be unstable. When a short input arrives to the neuron population, its activity therefore increases rapidly, using the window of opportunity before the inhibitory feedback current becomes prominent. Inhibition then takes over and largely terminates the excursion leading to a shorter duration impulse response of the network. For prolonged input, these impulses basically add up, which leads to the fast convergence to new stationary activity, amplifying the input. The complex Schur decomposition of the matrix governing the network dynamics reveals a strong feedforward coupling from an oscillatory difference to an oscillatory sum mode. This is similar to the networks in [21], which, however, have mostly real, non-positive eigenvalues and thus non-oscillatory modes without Hebbian amplification.

Balanced networks The balanced versions of our networks incorporate inhibition that closely tracks excitation with a short delay, as observed in experiments [60, 75, 76]. For stationary dynamics, excitatory and inhibitory currents largely cancel, resulting in a relatively weak net excitatory interaction between similarly-tuned neurons [18, 19]. However, excitation reacts slightly faster than inhibition to dynamic activity changes, allowing strong interactions during the window of opportunity where the change in excitation is not yet balanced by the delayed inhibition. This mechanism allows activity changes to propagate quickly through the network, significantly decreasing its response time.

The effective lag of inhibitory feedback originates in our SFA and in other balanced amplification [21] networks from the fact that the inhibitory currents are evoked by a low-pass filtered version of the excitatory activity. In contrast, in our balanced networks, there is an explicit, fixed lag between excitatory and inhibitory activity. We thus have an infinite dimensional dynamical system governed by a delay differential equation. However, the basic mechanism of shortening the impulse response and speeding up the reaction to inputs is the same: Excitation without feedback inhibition is unstable. When an input arrives, inhibition is insufficient to balance it. The excitatory activity therefore increases strongly during the temporal window of opportunity. Then inhibition sets in, largely balances excitation and limits it to its stationary value. We note that still excitation dominates our balanced networks. We have numerically and analytically studied the resulting convergence to stationary activity. This revealed qualitatively different types of dynamics, which are familiar from the harmonic oscillator, namely overdamped, critical and underdamped dynamics. Their occurrence depends on the strength of the inhibitory feedback and the size of the time lag between excitation and inhibition.

The increased response speed comes at the cost of requiring additional synapses and interneurons to implement the inhibition. In our model this roughly doubles the number of required synapses compared to the unbalanced cooperatively coding networks. As the number of needed synapses is still independent of the RF size, the balanced networks still save synapses for larger RFs. Furthermore, inhibitory neurons may be required anyways for purposes such as maintaining irregular spiking activity [55–58]. Alongside the increased synaptic requirements, the cancellation of currents means an increased metabolic cost.

We covered inhibition in a simplified, effective manner without explicitly incorporating inhibitory neurons. As argued above, we are optimistic that our qualitative findings will hold in more complex networks, for example if they are optimized for sparseness. In particular we expect that such networks will still show characteristics of cooperative coding and that inhibition will allow to reduce response times.

From our analysis, the brain might use cooperative coding to save synapses and space compared to a purely feedforward or more wasteful recurrent implementation, but

might invest some synapses, neurons, space and energy in balancing inhibition to retain a reasonable response speed.

Mixed selectivity We also considered networks with two-dimensional mixed selectivity, where neurons are not only selective to a single stimulus but to two stimuli. Since our networks are linear, we consider linear mixed selectivity. This is a simplification compared to the nonlinear mixed selectivity that is ubiquitous in the brain [64, 65]. We observe that the tradeoff between saved synapses and speed is for networks using linear mixed selectivity improved by a constant factor compared to networks where each neuron codes for a single stimulus.

Multi-dimensional stimuli We considered networks encoding one- and two-dimensional stimuli. We found that the trade-off between saved synapses and increased response time is much more favorable for neurons that encode 2D stimuli.

For $P \geq 2$-dimensional stimuli, the fact that activity simultaneously propagates along all dimensions suggests that the scaling of the response time with d, the characteristic RF width along one of the dimensions, does not change with the number of dimensions. However, the RF size n_{RF}, the number of synapses needed in a purely feedforward implementation, can be assumed to scale as $n_{\mathrm{RF}} \sim (2d+1)^P$, with a prefactor that depends on geometry. This reasoning suggests that the response time of networks encoding higher-dimensional stimuli scales like $\tau_{\mathrm{resp}} \sim n_{\mathrm{RF}}^{2/P}$ and $\tau_{\mathrm{resp}}^{\mathrm{bal}} \sim n_{\mathrm{RF}}^{1/P}$, which we verified for $P = 1, 2$. The number of synapses required in a cooperatively coding network will also in higher dimensions be negligible compared to n_{RF}. Therefore the number of saved synapses in the cooperatively coding network is still approximately n_{RF}. For higher-dimensional stimuli, the trade-off between response time and saved synapses would thus become highly beneficial: the response time τ_{resp} or $\tau_{\mathrm{resp}}^{\mathrm{bal}}$ would grow only slowly with the number of saved synapses n_{RF} due to the strongly sublinear relationship for larger P.

Properties of connectivity Our cooperative coding scheme relies on the presence of few strong recurrent excitatory connections between similarly-tuned cells. Excitation can be balanced by inhibition, while interactions between similarly-tuned cells remain net excitatory. Indeed, in layer 2/3 of mouse visual cortex, pyramidal neurons with the same orientation preference connect at higher rates and form more bidirectional connections [15]. This pattern of increased connection probability between neurons with highly similar tuning extends across layers and visual areas, including feedforward and feedback connections [14]. Furthermore, synapses between neurons with similar spatial RFs are markedly stronger such that neurons receive the majority of their local excitation from few similarly-tuned cells [16]. This strong, sparse local excitation matches the RF structure of the receiving neurons [16]. Recurrent connections are generally sparse in the cortex [77–80].

Studies of recurrent functional connectivity found net excitation between (spatially close) neurons with similar tuning [19] and most correlated responses [18], consistent with our model. Ref. [19] showed specifically that when optogenetically stimulating spatially compact ensembles of co-tuned neurons, similarly-tuned neurons were excited while differently-tuned neurons were inhibited. In a 1D-model, ref. [18] had to incorporate strong nearest-neighbor-like excitatory interactions to match the experimental data, arguing that they stabilize network responses in the presence of input noise. Our interpretation is that they not only stabilize, but actively form the RF. Refs. [18, 19] also show an inhibitory effect on largely differently tuned neurons. Ref. [18] found net inhibition between rather similarly tuned neurons as well. This is assumed to implement feature competition, which we did not include in our model.

A particular benefit of our cooperative coding scheme is that it allows feedforward connections to be sparse. This fits for example experimental observations in the primary visual cortex, where the vast majority of inputs are local recurrent ones, while only a few percent are feedforward inputs [81,82]. Ref. [83] estimated based on experimental studies [84] that a single hypercolumn in primate V1 receives only 10-30 feedforward inputs from the magnocellular layer of dorsal LGN mediating retinal input, with single cells in L4α receiving as little as 0 - 6 inputs. Also in the hippocampal region CA3, where place fields are enlarged compared to the upstream dentate gyrus [40], the recurrent connectivity is high [79,85].

Experiments that aim to disentangle feedforward from recurrent contributions to orientation selectivity resulted in mixed findings. Ref. [86] showed that the input to simple cells in L4 of cat primary visual cortex still exhibit tuned, modulated responses to drifting gratings after cortical activity was suppressed by cooling. In line with this, ref. [17] found that thalamic and cortical contributions to the first harmonic of the response curve (F1) were co-tuned. However, the temporally averaged response (F0) to drifting gratings was tuned only in cortical but not in thalamic inputs. A recent study, ref. [87], suggests that total input current from L4 of mouse primary visual cortex to L2/3 may lack orientation tuning and that orientation selectivity is determined by recurrent inputs from within L2/3.

Optimality By distributing and reusing computations via excitatory connections, cooperatively coding networks minimize or strongly reduce the number of synapses required to generate their responses. This corresponds to minimizing or strongly reducing the L0 norm of the weights (or synaptic currents).

In the following we compare the metabolic cost of the feedforward and the simplest, purely excitatory cooperative coding networks in the stationary state. We consider the three main contributions of the metabolic cost [26,28]: the cost of keeping up the resting network, of generating the required activity and of the synaptic transmissions. Keeping up the resting network (housekeeping and maintaining the resting potentials) requires the same energy expenditure in both implementations, as the number of neurons is the same. Also the energy required to generate the stationary output activity is the same, as corresponding neurons generate the same activity in both implementations.

The cost of synaptic transmission is dominated by the metabolic cost caused by the postsynaptic currents [28]. We assume that this cost is characterized by the sum of the absolute current strengths at individual synapses, i.e. by the L1 norm of the synaptic currents [88]. This is the same in both implementations. Finally, a small part of the overall cost (less than 10% [28]) arises due to presynaptic calcium influx and usage of neurotransmitter during synaptic transmission. A comparison of these contributions between the network implementations is more difficult. The surface of the active zone [89] increases linearly with the synaptic strength [90]. Assuming that the area where calcium influx happens increases linearly with the active zone size, the amount of influx during a single synaptic transmission depends linearly on the synaptic strength, consistent with [91]. The same should then hold for the related cost. Concerning the amount of neurotransmitter used, experiments observe that the number of directly releasable vesicles is linearly related to the synaptic strength, while the impact of a single vesicle and its release probability are independent of it [90,92] (this may differ for different connected neuron types [93]). The finding suggests a linear dependence between the synaptic strength and the used neurotransmitter and thus the related cost at a single transmission. The total cost due to presynaptic calcium influx and neurotransmitter usage is therefore proportional to the absolute synaptic strength times the number of transmissions (the presynaptic activity). This, in turn, is proportional to the absolute value of the induced postsynaptic current, which is identical in both implementations.

We conclude that the metabolic cost of generating the stationary state is the same in the feedforward and the simplest, purely excitatory cooperative coding implementation. Since the latter takes longer to reach the stationary state, however, it also requires more current and energy before generating a useful result. Due to the cancellation of excitatory and inhibitory currents, the networks with SFA and the balanced networks also use more current and thus more energy in the stationary state.

Previous studies often minimized the number of spikes or, more generally, the neuronal activity needed to represent encoded features. Refs. [30,32,34] follow this approach and suggest that tight EI-balance may be a signature of a highly coordinated and competitive code that, despite the irregular firing, is orders of magnitude more precise than a Poisson rate code. This spike-code depends on an extremely structured, dense connectivity, through which similarly-coding neurons quickly inhibit each other to prevent redundant spiking. From this standpoint the findings of excitatory functional connectivity between similarly-tuned neurons [15,16,18,19] seem counter-intuitive.

Although the energetic costs of spike generation and synaptic transmission very likely play a role in shaping cortical networks, our work suggests that space constraints and the number of required synapses may be another main factor.

Conclusion To conclude, net excitatory connectivity between similarly tuned neurons is compatible with a cooperative coding scheme that generates network responses with a minimal number of synapses. This suggests space constraints as an important factor in shaping neural networks. The window of opportunity between excitation and balancing, delayed adaptation or inhibition may be harnessed to rapidly propagate activity changes through the network, speeding up equilibration times by orders of magnitude.

Methods

All simulations have periodic boundary conditions. Fixed network parameters are the number of neurons N for 1D and N^2 for 2D networks, the neuronal time-constant τ and, in networks with inhibition, the EI-lag τ_{lag}. We set $N = 200$, $\tau = 1$ and $\tau_{\text{lag}} = 0.1$. In the networks with SFA we use a fixed value of $\tau_{\text{SFA}} = \tau = 1$ and, for each RF size, obtain the value of a_{SFA} that minimizes the temporal mean of the normalized L1-loss, $(1/T)\int_0^T \mathrm{d}t\ |x(t) - x^*(t)|_1/|x^*|_1$, through a linear grid search. Here $T = 500\tau$ is the length of a trial as described in Fig. 8a) and $x^*(t)$ the target corresponding to the present input. Fig. 8e) and c) show the scans over a_{SFA} and the individual loss curves for the optimal a_{SFA} values. In all networks with inhibition, we set $w_{\text{sum}}^{\text{rec,I}}$ to its critical value given by eq. (77).

We simulate our networks with SFA using the Euler method and all other networks using the midpoint method with stepsize $\mathrm{d}t = 0.01$. To simulate the networks with delayed inhibition, we also need midpoint values of the delayed activity. We obtain them by copying the midpoint values of the non-delayed activity τ_{lag} ($\tau_{\text{lag}}/\,\mathrm{d}t$ simulation steps) before.

For the data in Figs. 4 and 6, we obtain RFs with different sizes by setting $w_{\text{sum}}^{\text{rec}}$ or $w_{\text{sum}}^{\text{rec,net}}$ to appropriate values. In the case of 1D networks with and without SFA and in the case of 2D linear MS networks, we have analytical expressions for the RF sizes as a function of $w_{\text{sum}}^{\text{rec}}$ or $w_{\text{sum}}^{\text{rec,net}}$. We thus chose $w_{\text{sum}}^{\text{rec}}$ or $w_{\text{sum}}^{\text{rec,net}}$ such that the RF sizes are sampled linearly from $n_{\text{RF}}^{\text{1D}} = 6$ to $n_{\text{RF}}^{\text{1D}} = 50$ in steps of two. For the 2D network, we simulate networks with 20 different values of $w_{\text{sum}}^{\text{rec}}$ or $w_{\text{sum}}^{\text{rec,net}}$ and measure the RF sizes that the networks generate after convergence. We obtain $w_{\text{sum}}^{\text{rec}}$ or $w_{\text{sum}}^{\text{rec,net}}$ as $w_{\text{sum}}^{\text{rec}}$(or $w_{\text{sum}}^{\text{rec,net}}$) $= 1 - \tau/\tau_{\text{resp}}$ by varying τ_{resp} from 10 to 1,000 with equal spacing on a logarithmic scale.

To numerically determine a network's response time, we first simulate the network for a long time, clearly longer than the convergence time, and define the resulting state as the final, target state x^*. The loss is the L1 norm of the difference between x^* and the current state. We then simulate the network for a second time. We obtain τ_{resp} or $\tau_{\text{resp}}^{\text{bal}}$ as the earliest time at which the loss drops and stays below e^{-1} times the initial loss.

Acknowledgments

We thank Felipe Kalle Kossio for fruitful discussions.

References

1. Dayan P, Abbott LF. Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems. Computational Neuroscience. Massachusetts Institute of Technology Press; 2001.

2. Trappenberg T. Fundamentals of Computational Neuroscience. OUP Oxford; 2010. Available from: https://books.google.de/books?id=4PDsA1EVCx0C.

3. Hubel DH, Wiesel TN. Receptive Fields, Binocular Interaction and Functional Architecture in the Cat's Visual Cortex. The Journal of Physiology. 1962;160(1):106–154. doi:10.1113/jphysiol.1962.sp006837.

4. Nauhaus I, Nielsen KJ, Callaway EM. Efficient Receptive Field Tiling in Primate V1. Neuron. 2016;91(4):893–904. doi:10.1016/j.neuron.2016.07.015.

5. Taube S. Head Direction Cells Recorded in the Anterior Thalamic Nuclei of Freely Moving Rats. The Journal of Neuroscience. 1995;.

6. DiCarlo JJ, Johnson KO, Hsiao SS. Structure of Receptive Fields in Area 3b of Primary Somatosensory Cortex in the Alert Monkey. The Journal of Neuroscience. 1998;18(7):2626–2645. doi:10.1523/JNEUROSCI.18-07-02626.1998.

7. O'Keefe J, Dostrovsky J. The Hippocampus as a Spatial Map. Preliminary Evidence from Unit Activity in the Freely-Moving Rat. Brain Research. 1971;34(1):171–175. doi:10.1016/0006-8993(71)90358-1.

8. Hafting T, Fyhn M, Molden S, Moser MB, Moser EI. Microstructure of a Spatial Map in the Entorhinal Cortex. Nature. 2005;436(7052):801–806. doi:10.1038/nature03721.

9. Stensola H, Stensola T, Solstad T, Frøland K, Moser MB, Moser EI. The Entorhinal Grid Map Is Discretized. Nature. 2012;492(7427):72–78. doi:10.1038/nature11649.

10. Fukushima K. Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position. Biological Cybernetics. 1980;36(4):193–202. doi:10.1007/BF00344251.

11. Olshausen BA, Field DJ. Emergence of Simple-Cell Receptive Field Properties by Learning a Sparse Code for Natural Images. Nature. 1996;381(6583):607–609. doi:10.1038/381607a0.

12. Banino A, Barry C, Uria B, Blundell C, Lillicrap T, Mirowski P, et al. Vector-Based Navigation Using Grid-like Representations in Artificial Agents. Nature. 2018;557(7705):429–433. doi:10.1038/s41586-018-0102-6.

13. Cueva CJ, Wei XX. Emergence of grid-like representations by training recurrent neural networks to perform spatial localization. In: International Conference on Learning Representations; 2018.Available from: https://openreview.net/forum?id=B17JTOe0-.

14. Ding Z, Fahey PG, Papadopoulos S, Wang EY, Celii B, Papadopoulos C, et al.. type [; 2023]Available from: http://biorxiv.org/lookup/doi/10.1101/2023.03.13.531369.

15. Ko H, Hofer SB, Pichler B, Buchanan KA, Sjöström PJ, Mrsic-Flogel TD. Functional Specificity of Local Synaptic Connections in Neocortical Networks. Nature. 2011;473(7345):87–91. doi:10.1038/nature09880.

16. Cossell L, Iacaruso MF, Muir DR, Houlton R, Sader EN, Ko H, et al. Functional Organization of Excitatory Synaptic Strength in Primary Visual Cortex. Nature. 2015;518(7539):399–403. doi:10.1038/nature14182.

17. Lien AD, Scanziani M. Tuned Thalamic Excitation Is Amplified by Visual Cortical Circuits. Nature Neuroscience. 2013;16(9):1315–1323. doi:10.1038/nn.3488.

18. Chettih SN, Harvey CD. Single-Neuron Perturbations Reveal Feature-Specific Competition in V1. Nature. 2019;567(7748):334–340. doi:10.1038/s41586-019-0997-6.

19. Oldenburg IA, Hendricks WD, Handy G, Shamardani K, Bounds HA, Doiron B, et al. The Logic of Recurrent Circuits in the Primary Visual Cortex. Nature Neuroscience. 2024;27(1):137–147. doi:10.1038/s41593-023-01510-5.

20. Valente M, Pica G, Bondanelli G, Moroni M, Runyan CA, Morcos AS, et al. Correlations Enhance the Behavioral Readout of Neural Population Activity in Association Cortex. Nature Neuroscience. 2021;24(7):975–986. doi:10.1038/s41593-021-00845-1.

21. Murphy BK, Miller KD. Balanced Amplification: A New Mechanism of Selective Amplification of Neural Activity Patterns. Neuron. 2009;61(4):635–648. doi:10.1016/j.neuron.2009.02.005.

22. Lim S, Goldman MS. Balanced Cortical Microcircuitry for Maintaining Information in Working Memory. Nature Neuroscience. 2013;16(9):1306–1314. doi:10.1038/nn.3492.

23. Ben-Yishai R, Bar-Or RL, Sompolinsky H. Theory of Orientation Tuning in Visual Cortex. Proceedings of the National Academy of Sciences. 1995;92(9):3844–3848. doi:10.1073/pnas.92.9.3844.

24. Miller P. A Recurrent Network Model of Somatosensory Parametric Working Memory in the Prefrontal Cortex. Cerebral Cortex. 2003;13(11):1208–1218. doi:10.1093/cercor/bhg101.

25. Zhang WH, Wu S, Josić K, Doiron B. Sampling-Based Bayesian Inference in Recurrent Circuits of Stochastic Spiking Neurons. Nature Communications. 2023;14(1):7074. doi:10.1038/s41467-023-41743-3.

26. Attwell D, Laughlin SB. An Energy Budget for Signaling in the Grey Matter of the Brain. Journal of Cerebral Blood Flow & Metabolism. 2001;21(10):1133–1145. doi:10.1097/00004647-200110000-00001.

27. Zalc B. The Acquisition of Myelin: A Success Story. In: Chadwick DJ, Goode J, editors. Novartis Foundation Symposia. vol. 276. 1st ed. Wiley; 2006. p. 15–25. Available from: https://onlinelibrary.wiley.com/doi/10.1002/9780470032244.ch3.

28. Howarth C, Gleeson P, Attwell D. Updated Energy Budgets for Neural Computation in the Neocortex and Cerebellum. Journal of Cerebral Blood Flow & Metabolism. 2012;32(7):1222–1232. doi:10.1038/jcbfm.2012.35.

29. Garcia KE, Kroenke CD, Bayly PV. Mechanics of Cortical Folding: Stress, Growth and Stability. Philosophical Transactions of the Royal Society B: Biological Sciences. 2018;373(1759):20170321. doi:10.1098/rstb.2017.0321.

30. Boerlin M, Machens CK, Denève S. Predictive Coding of Dynamical Variables in Balanced Spiking Networks. PLoS Computational Biology. 2013;9(11):e1003258. doi:10.1371/journal.pcbi.1003258.

31. Thalmeier D, Uhlmann M, Kappen HJ, Memmesheimer RM. Learning universal computations with spikes. PLOS Comput Biol. 2016;12:e1004895.

32. Denève S, Machens CK. Efficient Codes and Balanced Networks. Nature Neuroscience. 2016;19(3):375–382. doi:10.1038/nn.4243.

33. Abbott LF, DePasquale B, Memmesheimer RM. Building Functional Networks of Spiking Model Neurons. Nature Neuroscience. 2016;19:350–355.

34. Kadmon J, Timcheck J, Ganguli S. Predictive Coding in Balanced Neural Networks with Noise, Chaos and Delays. NeurIPS Proceedings. 2020; p. 12.

35. Pulido C, Ryan TA. Synaptic Vesicle Pools Are a Major Hidden Resting Metabolic Burden of Nerve Terminals. SCIENCE ADVANCES. 2021;.

36. Jones JP, Palmer LA. An Evaluation of the Two-Dimensional Gabor Filter Model of Simple Receptive Fields in Cat Striate Cortex. Journal of Neurophysiology. 1987;58(6):1233–1258. doi:10.1152/jn.1987.58.6.1233.

37. Dumoulin SO, Wandell BA. Population Receptive Field Estimates in Human Visual Cortex. NeuroImage. 2008;39(2):647–660. doi:10.1016/j.neuroimage.2007.09.034.

38. Wandell BA, Winawer J. Computational Neuroimaging and Population Receptive Fields. Trends in Cognitive Sciences. 2015;19(6):349–357. doi:10.1016/j.tics.2015.03.009.

39. Serre T. In: Jaeger D, Jung R, editors. Hierarchical Models of the Visual System. New York, NY: Springer New York; 2013. p. 1–12. Available from: https://doi.org/10.1007/978-1-4614-7320-6_345-1.

40. Neher T, Azizi AH, Cheng S. From grid cells to place cells with realistic field sizes. PLoS One. 2017;12(7):e0181618.

41. Dayan P, Abbott L. Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems. Cambridge: MIT Press; 2001.

42. Gerstner W, Kistler WM, Naud R, Paninski L. Neuronal Dynamics - From single neurons to networks and models of cognition. Cambridge: Cambridge University Press; 2014.

43. Jetschke G. Mathematik der Selbstorganisation. Frankfurt am Main: Harri Deutsch; 2009.

44. Hirsch MW, Smale S. Differential equations, dynamical systems, and linear algebra. No. 60 in Pure and applied mathematics. San Diego [u.a.]: Acad. Press; 1974. Available from: http://gso.gbv.de/DB=2.1/CMD?ACT=SRCHA&SRT=YOP&IKT=1016&TRM=ppn+022705937&sourceid=fbw_bibsonomy.

45. Pernice V, Staude B, Cardanobile S, Rotter S. How structure determines correlations in neuronal networks. PLOS Comput Biol. 2011;7:e1002059.

46. Kalle Kossio YF, Goedeke S, van den Akker B, Ibarz B, Memmesheimer RM. Growing Critical: Self-Organized Criticality in a Developing Neural System. Physical Review Letters. 2018;121:058301. doi:10.1103/PhysRevLett.121.058301.

47. White OL, Lee DD, Sompolinsky H. Short-term memory in orthogonal neural networks. Phys Rev Lett. 2004;92:148102.

48. Ganguli S, Huh D, Sompolinsky H. Memory traces in dynamical systems. Proc Natl Acad Sci U S A. 2008;105:18970–18975. doi:10.1073/pnas.0804451105.

49. Izhikevich EM. Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting. Cambridge: MIT Press; 2007.

50. Gutkin B, Zeldenrust F. Spike Frequency Adaptation. Scholarpedia. 2014;9(2):30643. doi:10.4249/scholarpedia.30643.

51. Treves A. Mean-field analysis of neuronal spike dynamics. Network: Computation in Neural Systems. 1993;4(3):259–284. doi:10.1088/0954-898x_4_3_002.

52. Hansel D, Sompolinsky H. Modeling feature selectivity in local cortical circuits. In: Koch C, Segev I, editors. Methods in Neuronal Modeling, 2nd ed. Cambridge, MA.: MIT press, Cambridge, MA.; 1998. p. 499–566.

53. Fuhrmann G, Markram H, Tsodyks M. Spike Frequency Adaptation and Neocortical Rhythms. Journal of Neurophysiology. 2002;88(2):761–770. doi:10.1152/jn.2002.88.2.761.

54. Dehghani N, Peyrache A, Telenczuk B, Le Van Quyen M, Halgren E, Cash SS, et al. Dynamic Balance of Excitation and Inhibition in Human and Monkey Neocortex. Scientific Reports. 2016;6(1):23176. doi:10.1038/srep23176.

55. Gerstein G, Mandelbrot B. Random Walk Models for the Spike Activity of a Single Neuron. Biophys J. 1964;4:41–68.

56. Shadlen M, Newsome W. The variable discharge of cortical neurons: Implications for connectivity, computation, and information coding. J Neurosci. 1998;18:3870–3896.

57. Van Vreeswijk C, Sompolinsky H. Chaos in Neuronal Networks with Balanced Excitatory and Inhibitory Activity. Science. 1996;274(5293):1724–1726. doi:10.1126/science.274.5293.1724.

58. Renart A, de la Rocha J, Bartho P, Hollender L, Parga N, Reyes A, et al. The Asynchronous State in Cortical Circuits. Science. 2010;327:587–590.

59. Herstel LJ, Wierenga CJ. Network control through coordinated inhibition. Current Opinion in Neurobiology. 2021;67:34–41. doi:10.1016/j.conb.2020.08.001.

60. Okun M, Lampl I. Instantaneous Correlation of Excitation and Inhibition during Ongoing and Sensory-Evoked Activities. Nature Neuroscience. 2008;11(5):535–537. doi:10.1038/nn.2105.

61. Znamenskiy P, Kim MH, Muir DR, Iacaruso MF, Hofer SB, Mrsic-Flogel TD. Functional Specificity of Recurrent Inhibition in Visual Cortex. Neuron. 2024; p. S0896627323009728. doi:10.1016/j.neuron.2023.12.013.

62. Runyan CA, Schummers J, Van Wart A, Kuhlman SJ, Wilson NR, Huang ZJ, et al. Response features of parvalbumin-expressing interneurons suggest precise roles for subtypes of inhibition in visual cortex. Neuron. 2010;67(5):847–857.

63. Najafi F, Elsayed GF, Cao R, Pnevmatikakis E, Latham PE, Cunningham JP, et al. Excitatory and inhibitory subnetworks are equally selective during decision-making and emerge simultaneously during learning. Neuron. 2020;105(1):165–179.e8.

64. Rigotti M, Barak O, Warden MR, Wang XJ, Daw ND, Miller EK, et al. The Importance of Mixed Selectivity in Complex Cognitive Tasks. Nature. 2013;497(7451):585–590. doi:10.1038/nature12160.

65. Fusi S, Miller EK, Rigotti M. Why Neurons Mix: High Dimensionality for Higher Cognition. Current Opinion in Neurobiology. 2016;37:66–74. doi:10.1016/j.conb.2016.01.010.

66. McNaughton BL, Battaglia FP, Jensen O, Moser EI, Moser MB. Path integration and the neural basis of the 'cognitive map'. Nat Rev Neurosci. 2006;7(8):663–678.

67. Yang ZH, Chu YM. On Approximating the Modified Bessel Function of the Second Kind. Journal of Inequalities and Applications. 2017;2017(1):41. doi:10.1186/s13660-017-1317-z.

68. von der Malsburg C. Self-organization of orientation sensitive cells in the striate cortex. Kybernetik. 1973;14(2):85–100.

69. Redish AD, Elga AN, Touretzky DS. A coupled attractor model of the rodent head direction system. Network. 1996;7(4):671–685.

70. Renart A, Song P, Wang XJ. Robust spatial working memory through homeostatic synaptic scaling in heterogeneous cortical networks. Neuron. 2003;38:473–485.

71. Kushnir L, Fusi S. Neural Classifiers with Limited Connectivity and Recurrent Readouts. The Journal of Neuroscience. 2018;38(46):9900–9924. doi:10.1523/JNEUROSCI.3506-17.2018.

72. Turcu D, Abbott LF. Sparse RNNs Can Support High-Capacity Classification. PLOS Computational Biology. 2022;18(12):e1010759. doi:10.1371/journal.pcbi.1010759.

73. Chenkov N, Sprekeler H, Kempter R. Memory Replay in Balanced Recurrent Networks. PLOS Computational Biology. 2017;13(1):e1005359. doi:10.1371/journal.pcbi.1005359.

74. Nayebi A, Sagastuy-Brena J, Bear DM, Kar K, Kubilius J, Ganguli S, et al. Recurrent Connections in the Primate Ventral Visual Stream Mediate a Trade-Off Between Task Performance and Network Size During Core Object Recognition. Neural Computation. 2022;34(8):1652–1675. doi:10.1162/neco_a_01506.

75. Graupner M, Reyes AD. Synaptic Input Correlations Leading to Membrane Potential Decorrelation of Spontaneous Activity in Cortex. The Journal of Neuroscience. 2013;33(38):15075–15085. doi:10.1523/JNEUROSCI.0347-13.2013.

76. Barral J, Reyes AD. Synaptic Scaling Rule Preserves Excitatory–Inhibitory Balance and Salient Neuronal Network Dynamics. Nature Neuroscience. 2016;19(12):1690–1696. doi:10.1038/nn.4415.

77. Braitenberg V, Schüz A. Cortex: Statistics and geometry of neuronal connectivity. Springer Berlin; 1998.

78. Holmgren C, Harkany T, Svennenfors B, Zilberter Y. Pyramidal cell communication within local networks in layer 2/3 of rat neocortex. J Physiol. 2003;551.1:139–153.

79. Cutsuridis V, Graham BP, Cobb S, Vida I, editors. Hippocampal microcircuits. 2nd ed. Springer Series in Computational Neuroscience. Cham, Switzerland: Springer International Publishing; 2018.

80. Seeman SC, Campagnola L, Davoudian PA, Hoggarth A, Hage TA, Bosma-Moody A, et al. Sparse recurrent excitatory connectivity in the microcircuit of the adult mouse and human cortex. Elife. 2018;7.

81. da Costa NM, Martin KAC. The proportion of synapses formed by the axons of the lateral geniculate nucleus in layer 4 of area 17 of the cat. J Comp Neurol. 2009;516(4):264–276.

82. Markov NT, Misery P, Falchier A, Lamy C, Vezoli J, Quilodran R, et al. Weight consistency specifies regularities of macaque cortical networks. Cereb Cortex. 2011;21(6):1254–1272.

83. Chariker L, Shapley R, Young LS. Orientation Selectivity from Very Sparse LGN Inputs in a Comprehensive Model of Macaque V1 Cortex. The Journal of Neuroscience. 2016;36(49):12368–12384. doi:10.1523/JNEUROSCI.2603-16.2016.

84. Angelucci A, Sainsbury K. Contribution of Feedforward Thalamic Afferents and Corticogeniculate Feedback to the Spatial Summation Area of Macaque V1 and LGN. Journal of Comparative Neurology. 2006;498(3):330–351. doi:10.1002/cne.21060.

85. Sammons RP, Vezir M, Moreno-Velasquez L, Cano G, Orlando M, Sievers M, et al. Structure and function of the hippocampal CA3 module. Proceedings of the National Academy of Sciences. 2024;121(6). doi:10.1073/pnas.2312281120.

86. Ferster D, Chung S, Wheat H. Orientation Selectivity of Thalamic Input to Simple Cells of Cat Visual Cortex. Nature. 1996;380(6571):249–252. doi:10.1038/380249a0.

87. Biggiogera J, Sanzeni A. Feature Tuning and Network Dynamics in Mouse Visual Cortex: Insights from Connectomics; 2023. Available from: http://doi.org/10.12751/nncn.bc2023.217.

88. Sacramento J, Wichert A, Van Rossum MCW. Energy Efficient Sparse Connectivity from Imbalanced Synaptic Plasticity Rules. PLOS Computational Biology. 2015;11(6):e1004265. doi:10.1371/journal.pcbi.1004265.

89. Welzel O, Henkel AW, Stroebel AM, Jung J, Tischbirek CH, Ebert K, et al. Systematic Heterogeneity of Fractional Vesicle Pool Sizes and Release Rates of Hippocampal Synapses. Biophysical Journal. 2011;100(3):593–601. doi:10.1016/j.bpj.2010.12.3706.

90. Holler S, Köstinger G, Martin KAC, Schuhknecht GFP, Stratford KJ. Structure and Function of a Neocortical Synapse. Nature. 2021;591(7848):111–116. doi:10.1038/s41586-020-03134-2.

91. Kirischuk S, Grantyn R. Inter-Bouton Variability of Synaptic Strength Correlates With Heterogeneity of Presynaptic Ca $^{2+}$ Signals. Journal of Neurophysiology. 2002;88(4):2172–2176. doi:10.1152/jn.2002.88.4.2172.

92. Loebel A. Multiquantal Release Underlies the Distribution of Synaptic Efficacies in the Neocortex. Frontiers in Computational Neuroscience. 2009;3. doi:10.3389/neuro.10.027.2009.

93. Silver RA, Lübke J, Sakmann B, Feldmeyer D. High-Probability Uniquantal Transmission at Excitatory Synapses in Barrel Cortex. Science. 2003;302(5652):1981–1984. doi:10.1126/science.1087160.

94. Gray RM. Toeplitz and Circulant Matrices: A Review. Foundations and Trends in Communications and Information Theory. 2006;2(3):155–239.

95. Corless RM, Gonnet GH, Hare DEG, Jeffrey DJ, Knuth DE. On the LambertW Function. Advances in Computational Mathematics. 1996;5(1):329–359. doi:10.1007/BF02124750.

Supporting information

S1 Dynamics of excitatory networks

The dynamics of our networks without SFA and explicit inhibition read for constant input $r_i(t) = r_i$

$$\tau \dot{x}_i(t) = -x_i(t) + \sum_j W_{ij}^{\text{rec}} x_j(t) + \sum_j W_{ij}^{\text{ff}} r_j, \tag{38}$$

see eq. (4). In the following, we discuss stationary states and time-dependent behaviors of these dynamics.

We first verify that the recurrent network eq. (5), where $W_{ij}^{\text{rec}} = (\delta_{i+1,j} + \delta_{i-1,j})/(\gamma + \gamma^{-1})$ and $W_{ij}^{\text{ff}} = \delta_{ij}\left(1 - 2\gamma/(\gamma + \gamma^{-1})\right)$, has the desired stationary state $x_i^{\text{resp}} =$

$\sum_j \gamma^{|i-j|} r_j$. For this we insert eq. (1) into eq. (5),

$$\tau \dot{x}_i = -\sum_j \gamma^{|i-j|} r_j + \frac{1}{\gamma + \gamma^{-1}} \left(\sum_j \gamma^{|i+1-j|} r_j + \sum_j \gamma^{|i-1-j|} r_j \right) + \left(1 - \frac{2\gamma}{\gamma + \gamma^{-1}} \right) r_i \tag{39}$$

$$= -\sum_j \gamma^{|i-j|} r_j + \frac{1}{\gamma + \gamma^{-1}} \left(\sum_{j<i+1} \gamma^{i+1-j} r_j + \sum_{j \geq i+1} \gamma^{j-i-1} r_j + \sum_{j \leq i-1} \gamma^{i-1-j} r_j + \sum_{j>i-1} \gamma^{j-i+1} r_j \right) + \left(1 - \frac{2\gamma}{\gamma + \gamma^{-1}} \right) r_i \tag{40}$$

$$= -\sum_j \gamma^{|i-j|} r_j + \frac{1}{\gamma + \gamma^{-1}} \left(\gamma \sum_{j \leq i} \gamma^{i-j} r_j + \gamma^{-1} \sum_{j>i} \gamma^{j-i} r_j + \gamma^{-1} \sum_{j<i} \gamma^{i-j} r_j + \gamma \sum_{j \geq i} \gamma^{j-i} r_j \right) + \left(1 - \frac{2\gamma}{\gamma + \gamma^{-1}} \right) r_i \tag{41}$$

$$= -\sum_j \gamma^{|i-j|} r_j + \frac{1}{\gamma + \gamma^{-1}} \left(\gamma \sum_j \gamma^{|i-j|} r_j + \gamma r_i + \gamma^{-1} \sum_j \gamma^{|i-j|} r_j - \gamma^{-1} r_i \right) + \frac{\gamma + \gamma^{-1} - 2\gamma}{\gamma + \gamma^{-1}} r_i \tag{42}$$

$$= \frac{\gamma - \gamma^{-1}}{\gamma + \gamma^{-1}} r_i + \frac{\gamma^{-1} - \gamma}{\gamma + \gamma^{-1}} r_i = 0 \tag{43}$$

The computation shows that the recurrent input from neurons $i+1$ and $i-1$ add, in the steady state, up to nearly generate the desired response of neuron i: the first and third summand in the bracket in eq. (42) give the desired response. The input that is missing, $\frac{\gamma - \gamma^{-1}}{\gamma + \gamma^{-1}} r_i < 0$, is contributed by the feedforward input (last summand in each line, which cancels the missing input in eq. (43)).

We now turn to non-stationary solutions. The recurrent weight matrices in our recurrent networks eq. (5) are real symmetric matrices and therefore diagonalizable with real eigenvalues and orthogonal eigenvectors. We denote by $\hat{x}^\mu$, $\mu = 0, \cdots, N-1$, the μth normalized eigenvector of W^{rec} with eigenvalue α^μ, and by $x^\mu(t) = \hat{x}^{\mu T} x(t)$ the projection of the network activity on this eigenvector (a scalar). Any activity $x(t)$ can be expressed as a linear combination of the orthonormal eigenvectors $\hat{x}^\mu$ with coefficients $x^\mu(t)$. $\hat{x}^\mu$ is also an eigenvector of $W^{\text{rec}} - \mathbb{1}$, with eigenvalue $\alpha^\mu - 1$. Multiplying eq. (38) with $\hat{x}^{\mu T}$ from the left shows that the evolution of network activity can be separated into the evolution of N individual components,

$$\tau \dot{x}^\mu(t) = -x^\mu(t) + \alpha^\mu x^\mu(t) + \sum_{i,j} \hat{x}_i^\mu W_{ij}^{\text{ff}} r_j, \tag{44}$$

$$\dot{x}^\mu(t) = -\lambda^\mu x^\mu(t) + \frac{1}{\tau} \sum_{i,j} \hat{x}_i^\mu W_{ij}^{\text{ff}} r_j, \tag{45}$$

see, e.g., [41]. Here we defined

$$\lambda^\mu = \frac{1 - \alpha^\mu}{\tau} \tag{46}$$

as the decay rate of network activity in the μth eigenmode. The response time of the network equals $1/\lambda^\mu$, if it is initialized in the μth eigenmode. When initialized by $x(0)$, the state decays to a stationary value in the different components,

$$x^\mu(t) = \mathrm{e}^{-\lambda^\mu t} x^\mu(0) + (1 - \mathrm{e}^{-\lambda^\mu t}) x^{*\mu}. \tag{47}$$

Here $x^\mu(0) = \hat{x}^{\mu T} x(0)$ are the initial and $x^{*\mu} = \hat{x}^{\mu T} x^* = \frac{1}{\lambda^\mu \tau} \sum_{i,j} \hat{x}_i^\mu W_{ij}^{\mathrm{ff}} r_j$ the final stationary values of the μth component in the eigenbasis. The vector of the stationary dynamics x^* has the entries $x_i^* = \sum_{j=1}^N \mathrm{RF}_{ij} r_j$, if the network generates the desired stationary dynamics. In the absence of recurrent input we have $W^{\mathrm{rec}} = 0$ and thus $\alpha^\mu = 0$, such that the activity decays at a rate of $\lambda = 1/\tau$ to its target. Positive eigenvalues $\alpha^\mu > 0$ mean a slower decay. At $\alpha^\mu = 1$ there is no decay at all, and for $\alpha^\mu > 1$ activity diverges. Stable activity thus requires the largest eigenvalue of the recurrent weight matrix to be smaller than one, such that together with the individual intrinsic decay of each neuron the dynamics are a contraction.

Because the system is linear, the different eigenmodes decay independently from each other at different rates, following eq. (47). Thus with time the faster-decaying modes connected to smaller eigenvalues become exponentially suppressed relative to the dominant, slowest-decaying mode.

The recurrent weight matrices in our recurrent networks eq. (5) furthermore have the property that they are circulant matrices, i.e. each row is equal to the row before rotated one element to the right. The eigenvalues of such matrices are given by the explicit formula [94]

$$\alpha^\mu = \sum_{j=1}^N W_{1j}^{\mathrm{rec}} \cdot \mathrm{e}^{\frac{2\pi I}{N}(j-1)\mu}, \tag{48}$$

where I is the imaginary unit. For purely excitatory networks, we have $W_{1j}^{\mathrm{rec}} \geq 0$, such that α^μ is maximal if $\mu = 0$, as otherwise the real part of each addend in eq. (48) is smaller or equal. The corresponding eigenvector of W^{rec} is

$$\hat{x}^0 = \frac{1}{\sqrt{N}}(1, 1, 1, ..., 1)^T. \tag{49}$$

Therefore, we obtain the slowest convergence for

$$\alpha^{\max} = \alpha^0 = w_{\mathrm{sum}}^{\mathrm{rec}} = \sum_{j=1}^N W_{ij}^{\mathrm{rec}}, \tag{50}$$

where $w_{\mathrm{sum}}^{\mathrm{rec}}$ is the row sum of the recurrent coupling matrix, which is independent of i as the matrix is circulant. The slowest exponential decay dominates the behavior for longer times. Inserting $\alpha^{\max}$ into eq. (46) thus yields τ_{resp}, the generic time scale of convergence of the network dynamics eq. (5) to the target state $x_i = \sum_j \mathrm{RF}_{ij} r_j$,

$$\tau_{\mathrm{resp}} = \frac{\tau}{1 - \alpha^0} = \frac{\tau}{1 - w_{\mathrm{sum}}^{\mathrm{rec}}}. \tag{51}$$

S2 Loss evolution of excitatory networks

In this section, we compute the time evolution of the network loss for our networks without SFA and explicit inhibition. We assume that the networks receive constant input and have initial state $x_i(0) = 0$. All convergence time scales that are present in the network (cf. eqs. (46) and (48)) could in principle contribute to this time evolution. We

will however see, that the decay time of the loss is equal to that of the slowest-decaying mode.

We define the loss as the 1-norm of the deviation of the network activity from the target activity,

$$L(t) = \frac{1}{N}\sum_{i=1}^{N} |x_i(t) - x_i^*|. \tag{52}$$

Under the assumption that network activity does not 'overshoot', i.e. that it is always lower than or equal to the target activity, $x_i(t) \leq x_i^* \ \forall t$, we can replace the loss function by a linear loss function

$$L^{\text{lin}}(t) = \frac{1}{N}\sum_{i=1}^{N} x_i^* - x_i(t). \tag{53}$$

We can express the linearized loss as the (scaled) projection of the deviation of the activity from its target, $x^* - x(t)$, onto the eigenvector $\hat{x}^0$ eq. (49),

$$L^{\text{lin}}(t) = \frac{1}{\sqrt{N}}\hat{x}^{0T}\left(x^* - x(t)\right) = \frac{1}{\sqrt{N}}\left(x^{*0} - x^0(t)\right). \tag{54}$$

The linear loss thus has a single exponential decay. This is a consequence of the fact that $(1, 1, 1, ...1)^T$ is an eigenvector of the dynamics, which, in turn, holds because the row sum of a circulant matrix is the same for each row. The linear loss therefore decays with the time constant τ_{resp} obtained in eq. (51) to its stationary value, 0.

S3 Cooperative coding minimizes the number of required synapses in 1D excitatory networks

The cooperatively coding recurrent networks eq. (5) and, with SFA, eq. (17) generate the required stationary dynamics eq. (1) with three synapses per neuron. Here we show that this is the minimal number of synapses. Specifically, we show that it is impossible to construct the RF of one neuron as the sum of only one or two other RFs and/or feedforward inputs.

Any network of the form eq. (4) that correctly implements the desired target response has RFs that satisfy eq. (8),

$$\text{RF}_{ij} = W_{ij}^{\text{ff}} + \sum_k W_{ik}^{\text{rec}}\text{RF}_{kj}, \tag{55}$$

i.e. $\text{RF}_{i\cdot}$ is a sum of localized peaks from feedforward inputs and extended two-sided exponentials from recurrent inputs (see Fig. 2c) for an example). First, there needs to be at least one (nonzero) feedforward synapse, as otherwise the network could not respond to input neuron activity. Clearly two feedforward synapses cannot solve the problem for RFs with $n_{\text{RF}} > 2$. But the sum of a localized peak from a feedforward input and the extended response from a recurrent input (with nonzero coefficients) cannot match the target shape (because the difference between the target RF and another RF is either zero or has extended support). Thus every neuron needs at least three synapses to implement the target network response.

S4 Spike frequency adaptation

Here we study the dynamics of a 1D network with SFA, which are given by eq. (17). Fig. 8a) shows the dynamic response of neuron j_0, whose preferred input is presented

as isolated unit input. The initial dynamics are faster than for a network without SFA. This is because the adaptation current effectively reduces the membrane time constant and because it does not yet fully compensate for the stronger recurrent and feedforward weights, as it is slow (see eqs. (14) and (16)).

To quantify the change in response speed, we compute the integrated loss, as a fast convergence implies that the integrated loss is small. We normalize the loss, $\text{Loss}(t) = |x(t) - x^*|_1/|x^*|_1$ to render it comparable for different RF widths. We define the integrated loss as the temporal mean of the normalized loss. Fig. 8d) shows the dependence of the integrated loss on the SFA time constant. The left-most data point at $\tau_{\text{SFA}} = 0$ corresponds to the zero-lag limit where $u_i(t) = x_i(t)$, which effectively modifies the leak current to $-(1 + a_{\text{SFA}})x_i(t)$ (eq. (16)), resulting in a reduced neuronal time constant $\tau \to \tau/(1 + a_{\text{SFA}})$. Consequently, in this limit and for $a_{\text{SFA}} = 1$, the response time of the SFA network is half as large as that of a network without SFA, which is $\tau_{\text{resp}} = \tau/(1 - w^{\text{rec}}_{\text{sum}}) \approx 201$, eq. (11). Also the integrated loss is smaller; without SFA it would be ≈ 0.413. The negative slope at $\tau_{\text{SFA}} = 0$ and the clear minimum at $\tau_{\text{SFA}} \approx 0.87$ show that the network can increase its response speed beyond this effective reduction of τ by utilizing the larger recurrent and feedforward weights, eq. (15), which are not yet fully compensated during equilibration. For oscillating dynamics, τ_{resp} can jump discontinuously. This happens if the time point at which all later losses are below e^{-1} jumps from one oscillation peak to the next. Therefore we use the integrated loss here as the minimization target.

Fig. 8f) shows the integrated loss as a function of a_{SFA} and $w^{\text{rec,I}}_{\text{sum}}$ in networks that combine SFA and balancing inhibition. The integrated loss is minimized in networks with critical inhibitory strength and without SFA. Larger adaptation can partly compensate weaker inhibition; the data suggest that the optimal values of the adaptation and inhibition parameters satisfy the relation $a_{\text{SFA}}\tau_{\text{SFA}} + w^{\text{rec,I}}_{\text{sum}}/w^{\text{rec,I}}_{\text{sum},c} \approx 1$.

S5 Effective residual inhibitory interaction strength

The residual inhibitory interaction term in the dynamics eq. (22) reads

$$\tau\dot{x}_i = \ldots - \sum_{j=1}^{N} W^{\text{rec,I}}_{ij}\Delta x_j(t) + \ldots . \tag{56}$$

In the following we compute the effective strength of the interaction mediated by this term, which we define as the total, integrated contribution to the state change of feature neuron i that it causes. For this we assume that the presynaptic activity x_j of neuron j changes only for a limited amount of time, i.e. its derivative has limited support. Concretely we assume that $\dot{x}_j(t) = 0$ for $|t| > T - \tau_{\text{lag}}$ for some finite time T. The component of the overall state change $\delta x_i = x_i(T) - x_i(-T)$ in neuron i between $-T$ and T that is caused by the residual inhibitory term due to changes in neuron j is then

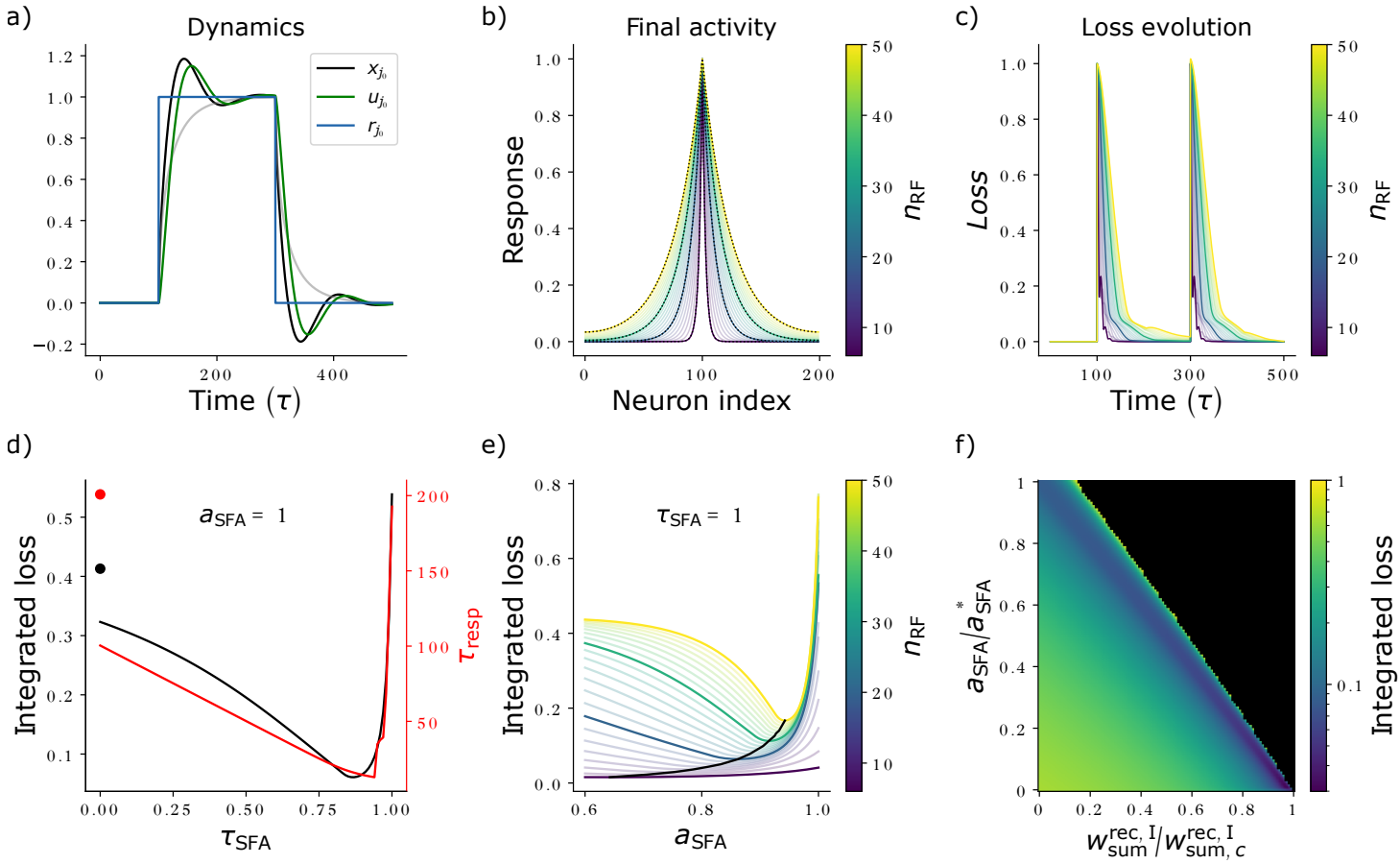

Fig 8: **Network and loss dynamics in a 1D network with SFA. a)** Dynamics of activity ($x_{j_0}(t)$, black), adaptation variable ($u_{j_0}(t)$, green) and input ($r_{j_0}(t)$, blue) for the $j_0 = 100$th neuron, which receives its preferred input as isolated unit input that is switched on at $t = 100\tau$ and off at $t = 300\tau$ ($r_j(t) = \delta_{jj_0}$ between these times). The network implements an RF with $d = 4.5$ ($n_{\mathrm{RF}} = 10$) and has, for better illustration, a rather slow $\tau_{\mathrm{SFA}} = 10\tau$ and slightly stronger-than-optimal $a_{\mathrm{SFA}} = 0.09$, resulting in a visible lag between $x(t)$ and $u(t)$ and oscillatory dynamics. During the initial rising phase ($t \gtrsim 100\tau$), activity rises faster than for a network without SFA (gray line). **b)** Stationary activity of SFA networks with different receptive field width. The networks receive an isolated input (parameters are $\tau_{\mathrm{SFA}} = \tau$ and optimal $a_{\mathrm{SFA}} = a^*_{\mathrm{SFA}}$, see e)). The stationary activity matches its target for different n_{RF} (color-coded). To show this, target (black dashed) and final activity (color-coded) of four networks with $n_{\mathrm{RF}} \in 6, 20, 34, 50$ are highlighted. **c)** Evolution of the L1-loss of the deviation of network activity from its target, normalized by the L1-norm of the target response for present input, under the stimulus protocol as described in a) for the same networks as in b), with a_{SFA} determined as described in e). **d)** Integrated loss (black), determined as the temporal mean of the normalized L1-loss shown in c), and response time (red), determined as the earliest time after the onset of input for which the normalized L1-loss drops and stays below e^{-1}, for a network with $d_{\mathrm{RF}} = 10$ ($n_{\mathrm{RF}} = 21$), $a_{\mathrm{SFA}} = 1$ and 101 values of τ_{SFA} scanned between 0 and 1. The integrated loss and response time of a network without SFA are shown by colored dots. **e)** Integrated loss for different n_{RF} (curves are color coded as in b)) as a function of a_{SFA} (for $\tau_{\mathrm{SFA}} = 1$). The minima, determining a^*_{SFA} used in b), c) and Fig. 6a), are connected by a black curve, showing that the optimal adaptation strength increases for larger RFs. We note that the data in a),d) and e) suggest that the optimal adaptation parameters fulfill the relation $a_{\mathrm{SFA}}\tau_{\mathrm{SFA}} \lesssim 1$. *(Caption continued on next page)*

Fig 8: *(Continued)* **f)** Grid scan of the integrated loss for a_{SFA} linearly scanned between 0 and a^*_{SFA} and $w^{\text{rec,I}}_{\text{sum}}$ linearly scanned between 0 and $w^{\text{rec,I}}_{\text{sum},c}$ (101 values each, $d_{\text{RF}} = 10$ ($n_{\text{RF}} = 21$), $\tau_{\text{SFA}} = 1$). The white region indicates parameters for which network activity diverges, identified by an integrated loss larger than 1, cf. also e). Here the loss was computed for the time window with present input only, more precisely for $t \in [100\tau - \mathrm{d}t, 300\tau]$, different from a)-e). For all simulations, we created data by simulating 1D networks with $N = 200$, $\tau = 1$ using the Euler method with step size $\mathrm{d}t = 0.01$; n_{RF}, τ_{SFA}, a_{SFA} and $w^{\text{rec,I}}_{\text{sum}}$ varied as described above.

given by

$$\begin{aligned}
\int_{-T}^{T} \mathrm{d}t\ \frac{-W^{\text{rec,I}}_{ij}}{\tau}\Delta x_j(t) &= \frac{-W^{\text{rec,I}}_{ij}}{\tau}\left(\int_{-T}^{T}\mathrm{d}t\ x_j(t) - \int_{-T}^{T}\mathrm{d}t\ x_j(t-\tau_{\text{lag}})\right)\\
&= \frac{-W^{\text{rec,I}}_{ij}}{\tau}\left(\int_{-T}^{T}\mathrm{d}t\ x_j(t) - \int_{-T-\tau_{\text{lag}}}^{T-\tau_{\text{lag}}}\mathrm{d}\tilde{t}\ x_j(\tilde{t})\right)\\
&= \frac{-W^{\text{rec,I}}_{ij}}{\tau}\left(\int_{T-\tau_{\text{lag}}}^{T}\mathrm{d}t\ x_j(t) - \int_{-T-\tau_{\text{lag}}}^{-T}\mathrm{d}t\ x_j(t)\right)\\
&= \frac{-W^{\text{rec,I}}_{ij}}{\tau}\Big(\tau_{\text{lag}}x_j(T) - \tau_{\text{lag}}x_j(-T)\Big)\\
&= -\frac{\tau_{\text{lag}}}{\tau}W^{\text{rec,I}}_{ij}\ \delta x_j. \qquad (57)
\end{aligned}$$

Here we substituted $\tilde{t} = t - \tau_{\text{lag}}$ in the second line; in the third line we used that large parts of the two integrals cancel and in the fourth line that $x_j(t)$ is constant between $T - \tau_{\text{lag}}$ and T as well as between $-T - \tau_{\text{lag}}$ and $-T$. $\delta x_j = x_j(T) - x_j(-T)$ is the total change in presynaptic activity. The result states that a change δx_j in presynaptic activity causes a total postsynaptic state change of $-\frac{\tau_{\text{lag}}}{\tau}W^{\text{rec,I}}_{ij}\ \delta x_j$. This may also be intuitively understood as follows: A step-like activity change in $x_j(t_0)$ by $\delta x_j(t_0)$ at time t_0 increases $\Delta x_j(t)$ by $\delta x_j(t_0)$ for all t with $t_0 \le t \le t_0 + \tau_{\text{lag}}$. According to eq. (56) it thus changes $\dot{x}_i(t)$ by $-1/\tau W^{\text{rec,I}}_{ij}\delta x_j(t_0)$ for a duration of τ_{lag}. The integrated effect is thus $-\tau_{\text{lag}}/\tau W^{\text{rec,I}}_{ij}\delta x_j(t_0)$. A continuous change of $x_j(t)$ may be seen as assembled of many small step-like ones, which together sum to δx_j and thus have the integrated effect eq. (57).

In the limit of small τ_{lag}, the residual inhibitory interaction term transmits the temporal derivative of the neuronal dynamics in excitatory manner, adding a contribution with the same sign to the temporal derivative of the postsynaptic neuron. To see this, we scale the inhibitory weights with τ_{lag} in such a way that the total, integrated effect of the residual inhibition due to a presynaptic activity change at t is independent of the lag between excitatory and inhibitory activity, τ_{lag}, i.e. we set $W^{\text{rec,I}}_{ij} = \frac{\tau}{\tau_{\text{lag}}}c_{ij}$ with constant c_{ij}, compare eq. (57). In the limit of short lag the delayed interaction term then becomes $\lim_{\tau_{\text{lag}}\to 0} -W^{\text{rec,I}}_{ij}\Delta x_j(t) = \lim_{\tau_{\text{lag}}\to 0} -\tau c_{ij}\Delta x_j(t)/\tau_{\text{lag}} = -\tau c_{ij}\dot{x}_j(t)$. The prefactor $-\tau c_{ij} > 0$ is positive, which renders the coupling excitatory.

S6 Loss evolution of balanced networks

To analytically estimate the evolution of the loss in our networks with balancing, delayed inhibition, we use again the linearized L1 loss eq. (53)), i.e. we assume again that the activity is always lower than or equal to the target activity. In the purely excitatory network, the linearized loss satisfied some simple dynamical equations, eq. (54) and

eq. (45). We will see in the following that this also holds in our balanced networks. To obtain the dynamical equation we compute the temporal derivative of the loss, using the linear time evolution of the activityeqs. (22) and (24). We assume that the input is constant, $r_j(t) = r_j$, and use the knowledge that for such input the dynamics converge to a stationary target state x_i^*. This yields

$$\begin{aligned}\tau \dot{L}^{\text{lin}}(t) &= \frac{\tau}{N} \sum_{i=1}^{N} \frac{\mathrm{d}}{\mathrm{d}t} \left(x_i^* - x_i(t)\right) \\ &= -\frac{1}{N} \sum_{i=1}^{N} \Big(- x_i(t) + \sum_{j=1}^{N} W_{ij}^{\text{rec,net}} x_j(t) \\ &\qquad - \sum_{j=1}^{N} W_{ij}^{\text{rec,I}} \Delta x_j(t) + \sum_{j=1}^{N} W_{ij}^{\text{ff}} r_j \Big). \end{aligned} \tag{58}$$

We now specialize the computation further by taking into account that in the cases of interest for us, W^{rec}, $W^{\text{rec,I}}$ and thus also $W^{\text{rec,net}}$, as well as W^{ff} are circulant matrices. This implies that the column sums $w_{\text{sum}}^{\text{rec,E}} = \sum_{i=1}^{N} W_{ij}^{\text{rec,E}}$, $w_{\text{sum}}^{\text{rec,I}} = \sum_{i=1}^{N} W_{ij}^{\text{rec,I}}$, $w_{\text{sum}}^{\text{rec,net}} = \sum_{i=1}^{N} W_{ij}^{\text{rec,net}}$ and $w_{\text{sum}}^{\text{ff}} = \sum_{i=1}^{N} W_{ij}^{\text{ff}}$ are independent of the column j. Equation (58) thus simplifies to

$$\begin{aligned}\tau \dot{L}^{\text{lin}}(t) &= \frac{1}{N} \sum_{i=1}^{N} x_i(t) - w_{\text{sum}}^{\text{rec,net}} \frac{1}{N} \sum_{j=1}^{N} x_j(t) \\ &\quad + w_{\text{sum}}^{\text{rec,I}} \frac{1}{N} \sum_{j=1}^{N} \Delta x_j(t) - w_{\text{sum}}^{\text{ff}} \frac{1}{N} \sum_{j=1}^{N} r_j. \end{aligned} \tag{59}$$

The networks that we want to track analytically start with $x_i(0) = 0$, such that the initial linear loss is $L^{\text{lin}}(0) = \sum_{i=1}^{N} x_i^* - 0$. To describe the network loss dynamics, we can thus use

$$\frac{1}{N} \sum_{i=1}^{N} x_i(t) = \frac{1}{N} \sum_{i=1}^{N} x_i^* - 0 - \left(x_i^* - x_i(t)\right) = L^{\text{lin}}(0) - L^{\text{lin}}(t). \tag{60}$$

An alike equation holds for $\frac{1}{N} \sum_{i=1}^{N} x_i(t - \tau_{\text{lag}})$, such that

$$\begin{aligned}\frac{1}{N} \sum_{i=1}^{N} \Delta x_i(t) &= \frac{1}{N} \sum_{i=1}^{N} x_i(t) - x_i(t - \tau_{\text{lag}}) \\ &= L^{\text{lin}}(0) - L^{\text{lin}}(t) - \left(L^{\text{lin}}(0) - L^{\text{lin}}(t - \tau_{\text{lag}})\right) \\ &= -\left(L^{\text{lin}}(t) - L^{\text{lin}}(t - \tau_{\text{lag}})\right) = -\Delta L^{\text{lin}}(t), \end{aligned} \tag{61}$$

where we introduced the abbreviation

$$\Delta L(t) = L^{\text{lin}}(t) - L^{\text{lin}}(t - \tau_{\text{lag}}) \tag{62}$$

for the difference between the current and the delayed loss. Inserting eq. (60) and eq. (61) into eq. (59) gives

$$\tau \dot{L}^{\text{lin}}(t) = \left(1 - w_{\text{sum}}^{\text{rec,net}}\right)\left(L^{\text{lin}}(0) - L^{\text{lin}}(t)\right) - w_{\text{sum}}^{\text{rec,I}} \Delta L^{\text{lin}}(t) - w_{\text{sum}}^{\text{ff}} \frac{1}{N} \sum_{j=1}^{N} r_j. \tag{63}$$

To eliminate the explicit occurrence of the inputs, we use that in the stationary state, which is reached for $t \to \infty$, we have $L^{\mathrm{lin}}(\infty) = 0$, $\dot{L}^{\mathrm{lin}}(\infty) = 0$ and $\Delta L^{\mathrm{lin}}(\infty) = 0$. For $t \to \infty$, eq. (63) thus shows that

$$0 = (1 - w_{\mathrm{sum}}^{\mathrm{rec,net}}) L^{\mathrm{lin}}(0) - w_{\mathrm{sum}}^{\mathrm{ff}} \frac{1}{N} \sum_{j=1}^{N} r_j. \tag{64}$$

Employing this in eq. (63) gives our final dynamical equation for the linearized loss in terms of the variable L^{lin} only,

$$\tau \dot{L}^{\mathrm{lin}}(t) = -\big(1 - w_{\mathrm{sum}}^{\mathrm{rec,net}}\big) L^{\mathrm{lin}}(t) - w_{\mathrm{sum}}^{\mathrm{rec,I}} \Delta L^{\mathrm{lin}}(t). \tag{65}$$

We solve this delay differential equation with the ansatz

$$L^{\mathrm{lin}}(t) = L^{\mathrm{lin}}(0) \exp(-\Omega t). \tag{66}$$

We note that

$$\Omega = \lambda + \mathrm{i}\omega \tag{67}$$

is generally complex. The ansatz implies

$$\begin{aligned} \dot{L}^{\mathrm{lin}}(t) &= -\Omega L^{\mathrm{lin}}(t), \\ \Delta L^{\mathrm{lin}}(t) &= L^{\mathrm{lin}}(t) - L^{\mathrm{lin}}(t - \tau_{\mathrm{lag}}) = \left(1 - \exp(\Omega \tau_{\mathrm{lag}})\right) L^{\mathrm{lin}}(t) \\ &= -\left(\exp(\Omega \tau_{\mathrm{lag}}) - 1\right) L^{\mathrm{lin}}(t). \end{aligned} \tag{68}$$

Inserting eq. (68) into eq. (65) and dividing by $-\tau L^{\mathrm{lin}}(t)$ yields

$$\begin{aligned} -\tau \Omega L^{\mathrm{lin}}(t) &= -(1 - w_{\mathrm{sum}}^{\mathrm{rec,net}}) L^{\mathrm{lin}}(t) + w_{\mathrm{sum}}^{\mathrm{rec,I}} \big(\exp(\Omega \tau_{\mathrm{lag}}) - 1\big) L^{\mathrm{lin}}(t), \\ \Omega &= \frac{1 - w_{\mathrm{sum}}^{\mathrm{rec,net}}}{\tau} - \frac{w_{\mathrm{sum}}^{\mathrm{rec,I}}}{\tau} \big(\exp(\Omega \tau_{\mathrm{lag}}) - 1\big). \end{aligned} \tag{69}$$

We see immediately that in the absence of inhibition, $w_{\mathrm{sum}}^{\mathrm{rec,I}} = 0$, we have $\Omega = \lambda = \frac{1 - w_{\mathrm{sum}}^{\mathrm{rec,net}}}{\tau} = \frac{1}{\tau_{\mathrm{resp}}}$ (cf. eq. (11)), such that the decay rate of the purely excitatory network eq. (4) is recovered, as it has to be. To solve eq. (69) in presence of inhibition, we rewrite it as

$$\Omega \tau_{\mathrm{lag}} = \frac{\tau_{\mathrm{lag}}}{\tau_{\mathrm{resp}}} - \frac{\tau_{\mathrm{lag}}}{\tau} w_{\mathrm{sum}}^{\mathrm{rec,I}} \big(\exp(\Omega \tau_{\mathrm{lag}}) - 1\big), \tag{70}$$

$$\left(\Omega \tau_{\mathrm{lag}} - \frac{\tau_{\mathrm{lag}}}{\tau_{\mathrm{resp}}} - \frac{\tau_{\mathrm{lag}}}{\tau} w_{\mathrm{sum}}^{\mathrm{rec,I}}\right) \exp(-\Omega \tau_{\mathrm{lag}}) = -\frac{\tau_{\mathrm{lag}}}{\tau} w_{\mathrm{sum}}^{\mathrm{rec,I}}. \tag{71}$$

Substituting

$$\Omega \tau_{\mathrm{lag}} - \frac{\tau_{\mathrm{lag}}}{\tau_{\mathrm{resp}}} - \frac{\tau_{\mathrm{lag}}}{\tau} w_{\mathrm{sum}}^{\mathrm{rec,I}} = -z, \tag{72}$$

we obtain

$$z \exp(z) = \frac{\tau_{\mathrm{lag}}}{\tau} w_{\mathrm{sum}}^{\mathrm{rec,I}} \exp\left(\frac{\tau_{\mathrm{lag}}}{\tau_{\mathrm{resp}}} + \frac{\tau_{\mathrm{lag}}}{\tau} w_{\mathrm{sum}}^{\mathrm{rec,I}}\right). \tag{73}$$

The branches of the Lambert W function solve $z \exp(z) = \mathrm{RHS}$ for z. Applying them to eq. (73) yields $z = W_k(\mathrm{RHS})$, where W_k denotes the kth branch and RHS the right hand side of eq. (73),

$$z = W_k\left(\frac{\tau_{\mathrm{lag}}}{\tau} w_{\mathrm{sum}}^{\mathrm{rec,I}} \exp\left(\frac{\tau_{\mathrm{lag}}}{\tau_{\mathrm{resp}}} + \frac{\tau_{\mathrm{lag}}}{\tau} w_{\mathrm{sum}}^{\mathrm{rec,I}}\right)\right). \tag{74}$$

Resubstituting for z then gives the decay rate λ (cf. eq. (67)) and, if an oscillation is present, the oscillation frequency ω of the linearized loss,

$$\Omega\tau_{\text{lag}} = \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}} + \frac{\tau_{\text{lag}}}{\tau} w_{\text{sum}}^{\text{rec,I}} - W_k\left(\frac{\tau_{\text{lag}}}{\tau} w_{\text{sum}}^{\text{rec,I}} \exp\left(\frac{\tau_{\text{lag}}}{\tau_{\text{resp}}} + \frac{\tau_{\text{lag}}}{\tau} w_{\text{sum}}^{\text{rec,I}}\right)\right). \tag{75}$$

Figs. 5 and 9 show the relevant solutions, obtained from the branches $k = 0$ and $k = -1$.

Similarly to the dynamics of a dampened harmonic oscillator, there is a transition from an exponentially-decaying ('overdamped') regime with two decay rates to an oscillating ('underdamped') regime with a single one. At the transition point we have critical inhibitory strength ('critical damping'). In the oscillatory regime it is the amplitude of the oscillation that decays exponentially. Because the actual error is always smaller or equal to the amplitude, having slightly larger than critical inhibition may be optimal, see Fig. 5a). The solutions corresponding to branches other than $k = 0, -1$ have markedly higher decay rates and are thus of little relevance. They also have high oscillation frequencies with periods smaller than the lag.

We note that in the case of oscillations ($\omega \neq 0$), the linearized loss eq. (66) periodically reaches zero. This corresponds to $x_i(t) = x_i^*$ for all neurons, meaning that all neurons simultaneously go from positive to negative deviations from their target activities, or vice-versa. In the full network model, however, these transitions will occur at different times, so that when one neuron matches its target activity there are others that don't. Therefore, the minima of the oscillation are not at zero but at a final error, see Fig. 5a).

We now determine the critical inhibitory strength and the critical decay rate. These are determined by $z = W_k(\text{RHS})$ (eq. (74), with RHS the right hand side of eq. (73)) having only one real solution z. This happens at the branch point $\text{RHS} = -\text{e}^{-1}$ where the 0th and -1st branch agree, $z = W_0(-\text{e}^{-1}) = W_{-1}(-\text{e}^{-1}) = -1$ [95]. We first set $\text{RHS} = z\text{e}^z = -\text{e}^{-1}$ to obtain $w_{\text{sum},c}^{\text{rec,I}}$,

$$\frac{\tau_{\text{lag}}}{\tau} w_{\text{sum},c}^{\text{rec,I}} \exp\left(\frac{\tau_{\text{lag}}}{\tau} w_{\text{sum},c}^{\text{rec,I}}\right) = -\exp\left(-1 - \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}}\right), \tag{76}$$

$$\frac{\tau_{\text{lag}}}{\tau} w_{\text{sum},c}^{\text{rec,I}} = W_k\left(-\exp\left(-1 - \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}}\right)\right), \tag{77}$$

where we have again used a (yet unspecified) branch of the Lambert W function to solve for $w_{\text{sum},c}^{\text{rec,I}}$. We know that the inhibitory strength must be real and negative; this can only hold for $k = 0$ or $k = -1$. It might seem surprising that we obtain two solutions here. The reason is that there are actually two critical points, seeFig. 9a): one with a positive decay rate, and another with a negative decay rate, which thus describes exponential growth. We are interested only in the converging dynamics and thus choose $k = 0$.

Next we resubstitute $\lambda^c\tau_{\text{lag}} = \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}} + \frac{\tau_{\text{lag}}}{\tau} w_{\text{sum},c}^{\text{rec,I}} - z$ (eq. (72) with Ω replaced by the critical, real decay rate λ^c), use $z = -1$ and insert the critical inhibitory strength eq. (77) to obtain the critical decay rate,

$$\lambda^c\tau_{\text{lag}} = \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}} + \frac{\tau_{\text{lag}}}{\tau} w_{\text{sum},c}^{\text{rec,I}} + 1 \tag{78}$$

$$= 1 + \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}} + W_0\left(-\exp\left(-1 - \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}}\right)\right). \tag{79}$$

The response time of the balanced network is the inverse of λ^c,

$$\tau_{\text{resp}}^{\text{bal},c} = \frac{1}{\frac{1}{\tau_{\text{lag}}} + \frac{1}{\tau_{\text{resp}}} + \frac{1}{\tau_{\text{lag}}} W_0\left(-\exp\left(-1 - \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}}\right)\right)}. \tag{80}$$

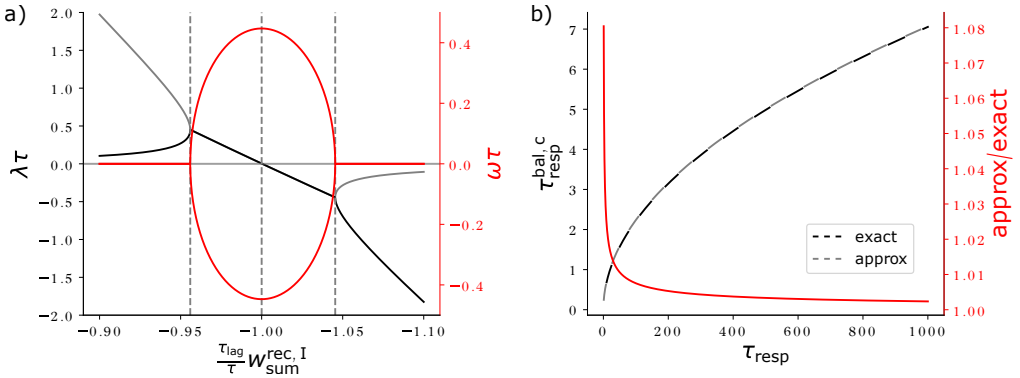

Fig 9: **Critical points and approximation of the analytical solution for $\tau^{\mathrm{bal,c}}_{\mathrm{resp}}$. a)** Real part (decay rate λ, black/gray) and imaginary part (oscillation frequency ω times ± 1, red) of the complex frequency of the exponential loss evolution, scaled by τ (compare Fig. 5). The leftmost dashed gray vertical line marks the first critical inhibitory strength at which there is only a single, positive decay rate. For $(\tau_{\mathrm{lag}}/\tau) w^{\mathrm{rec,I}}_{\mathrm{sum}} = -1$, marked by the middle dashed gray vertical line, the decay rate is zero and transitions from positive (decaying) to negative (exponentially growing). There is a second critical inhibitory strength, marked by the rightmost dashed gray vertical line, at which there is a single, negative decay rate. The critical point with the decaying dynamics corresponds to the solution of eq. (77) with $k = 0$, the one with the exponentially growing dynamics to that with $k = -1$. **b)** Exact (black dashed, cf. eq. (80)) and approximate (gray dashed, cf. eq. (89)) values of the critical decay time of the balanced network, $\tau^{\mathrm{bal,c}}_{\mathrm{resp}}$, as a function of that of the excitatory network. Approximation and exact solution agree well. Their ratio (red) is close to one, and approaches one for $\tau_{\mathrm{resp}} \gg \tau_{\mathrm{lag}}$. Parameters: $\tau = 1$, $\tau_{\mathrm{lag}} = 0.1$ and, in a), $w^{\mathrm{rec,net}}_{\mathrm{sum}} = 0.99$.

We now want to derive a more easily interpretable approximation for λ^c and $\tau_{\text{resp}}^{\text{bal},c}$ for small $\tau_{\text{lag}}/\tau_{\text{resp}}$. To this end, we modify eq. (79) to

$$\lambda^c \tau_{\text{lag}} - 1 - \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}} = W_0\left(-\exp\left(-1 - \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}}\right)\right) \tag{81}$$

$$\Rightarrow \quad \left(\lambda^c \tau_{\text{lag}} - 1 - \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}}\right) \exp\left(\lambda^c \tau_{\text{lag}} - 1 - \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}}\right) = -\exp\left(-1 - \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}}\right) \tag{82}$$

$$\Leftrightarrow \quad \left(\lambda^c \tau_{\text{lag}} - 1 - \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}}\right) \exp\left(\lambda^c \tau_{\text{lag}}\right) = -1. \tag{83}$$

Here we have applied the inverse of W_0, $z \to z\mathrm{e}^z$, to both sides, and multiplied with $\exp(-1 - \tau_{\text{lag}}/\tau_{\text{resp}})$. Next we assume that $\lambda^c \tau_{\text{lag}}$ is small, meaning that the response time is much larger than the E-I lag, expand the exponential up to the second power in λ^c and τ_{lag}, and solve for $\lambda^c \tau_{\text{lag}}$,

$$\left(\lambda^c \tau_{\text{lag}} - 1 - \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}}\right)\left(1 + \lambda^c \tau_{\text{lag}} + \frac{1}{2}(\lambda^c \tau_{\text{lag}})^2 + \dots\right) = -1 \tag{84}$$

$$\lambda^c \tau_{\text{lag}} - 1 - \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}} + (\lambda^c \tau_{\text{lag}})^2 - \lambda^c \tau_{\text{lag}} - \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}} \lambda^c \tau_{\text{lag}} - \frac{1}{2}(\lambda^c \tau_{\text{lag}})^2 \approx -1 \tag{85}$$

$$\frac{1}{2}(\lambda^c \tau_{\text{lag}})^2 - \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}}(\lambda^c \tau_{\text{lag}}) - \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}} \approx 0 \tag{86}$$

$$\lambda^c_{\pm} \tau_{\text{lag}} \approx \frac{\tau_{\text{lag}}}{\tau_{\text{resp}}} \pm \sqrt{\left(\frac{\tau_{\text{lag}}}{\tau_{\text{resp}}}\right)^2 + 2\frac{\tau_{\text{lag}}}{\tau_{\text{resp}}}}. \tag{87}$$

Here only the positive solution makes sense. Since the delayed inhibition speeds up responses, our assumption that $\lambda^c \tau_{\text{lag}}$ is small implies that also $\tau_{\text{lag}}/\tau_{\text{resp}}$ with the response time of the network without delayed inhibition is small. We can thus neglect in the radicand of eq. (87)'s RHS the quadratic term compared to the linear one. Compared to the resulting square root term we can neglect the first, linear RHS term, such that we obtain

$$\lambda^c \tau_{\text{lag}} \approx \sqrt{2\frac{\tau_{\text{lag}}}{\tau_{\text{resp}}}}. \tag{88}$$

The response time of the balanced network follows as the inverse of λ^c,

$$\tau_{\text{resp}}^{\text{bal},c} \approx \sqrt{\frac{\tau_{\text{resp}} \tau_{\text{lag}}}{2}}. \tag{89}$$

Fig. 9 shows that, despite the simple formula, the quality of the approximation is very good.

S7 Initial response of balanced networks

In the following we explain that the discrepancy between the analytical and numerical linearized loss evolution, which results in different response times (see Fig. 6a), c)), stems from different initializations of neuronal activity. For this, we first note that an exponential function eq. (66), $L^{\text{lin}}(t) = L^{\text{lin}}(0)\exp(-\Omega t)$, with Ω solving eq. (69) and arbitrary amplitude $L^{\text{lin}}(0)$, is an eigenmode of the time evolution of the linearized loss eq. (65), as the time evolution preserves its functional form. The results of the last section describe the loss evolution if the system is in such an eigenmode. However, in the main text simulations, networks are not initialized in an eigenmode: They are initialized with zero activity and respond to a sudden jump in input. This corresponds

to an initial state with $x(t) = 0$ for $t \leq 0$; in particular $\Delta x(0) = 0$. Therefore both the initial and earlier losses equal the maximal loss, $L(t) = L(-\tau_{\text{lag}}) = |x^*|_1/N$ for $t \leq 0$ and $\Delta L(0) = 0$, see eqs. (61) and (62) and Fig. 10b). In contrast, for the eigenmodes we have $\Delta L^{\text{lin}}(t) = -(\exp(\Omega\tau_{\text{lag}}) - 1)L^{\text{lin}}(t)$ for $t \leq 0$ (from eqs. (62) and (66)); especially $\Delta L^{\text{lin}}(0) < 0$. The loss is thus not initialized in an eigenmode; the decay rate of the loss converges to that of the slowliest decaying eigenmode over time.

For the chosen initial conditions the loss decay speeds up during this process, see Fig. 6c and Fig. 10a),b). This can be understood as follows: In our network simulations, activity initially increases, such that $\Delta x_i(t)$ is positive. The amplifying current term $-W^{\text{rec,I}}\Delta x(t)$ ($-W^{\text{rec,I}}_{ij} \geq 0$) in eq. (58) is thus positive, which causes a dynamical speed-up in the activity increase. Its sum is proportional to $-\Delta L(t)$, which is initially zero. The term $-w^{\text{rec,I}}_{\text{sum}}\Delta L^{\text{lin}}(t) \leq 0$ introduces the dynamical speed up evoked by $\Delta x_i(t)$ into the equation of the loss dynamics, where it causes a faster decay of the loss. As both $\Delta x_i(t)$ and $|\Delta L(t)|$ increase, the terms that generate the speed up in the activity and in the loss dynamics increase in absolute value. As a consequence, the activity increases faster and the loss decays faster. The inset in Fig. 10b) shows that $|\Delta L|$ increases (initially in absolute, later in relative terms) and approaches the same proportionality to L as for the slowliest decaying eigenmode, which is non-oscillatory in the displayed example. This means that the system settles in the eigenmode. The loss curves then become parallel lines on the logarithmic axes (extrapolation of Fig. 10a)).

This different initialization and the resulting initially lower impact of lagged interactions explain why the red data points in Fig. 6 show longer response times than expected from the theory.

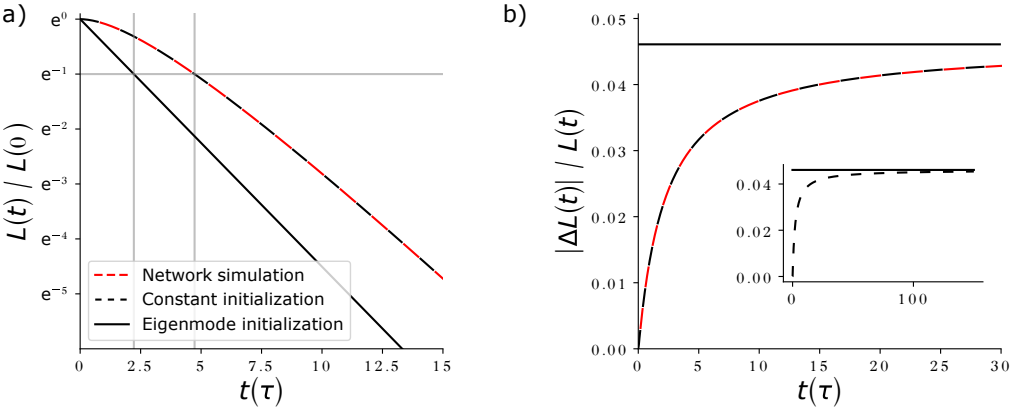

Fig 10: **Effects of initial conditions on loss evolution. a)** Normalized loss evolution obtained by solving eq. (65) (black curves) for critical inhibitory strength. The loss is either initialized in the slowest eigenmode (solid, analytical solution eqs. (66) and (69)), like in our analytical computations, or it is initialized as a constant function (dashed, $L(t) = L(0)$ for $t \leq 0$, numerical solution), like in our network simulations. Vertical lines indicate the time at which the error drops below e^{-1}, which experimentally defines the response time. The theory with constant initialization describes the loss evolution of a numerically simulated network (red dashed, $N = 200$) well. With time, the loss decay rate approaches the same value, independent of initialization; the curves become parallel. **b)** In the for the chosen parameters non-oscillatory slowliest decaying eigenmode, $\Delta L(t)$ is always proportional to $L(t)$ (solid horizontal line at the proportionality factor $1 - \mathrm{e}^{\lambda^c \tau_{\mathrm{lag}}} \approx -0.046$). With constant initialization (dashed curves), $\Delta L(t)$ is initially zero, such that the impact of the interactions mediated by $W^{\mathrm{rec,I}}$ (see eq. (24)) is initially small. $\Delta L(t)$ then tends to the same proportionality to $L(t)$ as for the eigenmode initialization. Inset: loss evolution for long times. Parameters: $w_{\mathrm{sum}}^{\mathrm{rec,net}} = 0.99$, $\tau = 1$, $\tau_{\mathrm{lag}} = 0.1$, $w^{\mathrm{rec,I}} = 0.5 w_{\mathrm{sum},c}^{\mathrm{rec,I}}$; we thus have $\frac{\tau_{\mathrm{lag}}}{\tau} w_{\mathrm{sum}}^{\mathrm{rec,I}} \approx -0.95594$; the displayed slowliest decaying eigenmode is non-oscillatory, cf. Fig. 9.

Bibliography

[1] P. Züge, C. Klos, and R.-M. Memmesheimer, *Weight versus Node Perturbation Learning in Temporally Extended Tasks: Weight Perturbation Often Performs Similarly or Better*, Physical Review X **13** (2023) 021006, ISSN: 2160-3308, (visited on 05/05/2023), © 2023 American Physical Society (cit. on pp. iii, 1, 3, 15, 20, 22, 24–26, 29, 30, 37, 41, 115, 175).

[2] P. Züge and R.-M. Memmesheimer, *Cooperative Coding of Continuous Variables in Networks with Sparsity Constraint*, 2024, eprint: https://www.biorxiv.org/content/10.1101/2024.05.13.593810v2, pre-published (cit. on pp. iii, 1, 3, 16, 18, 27, 33, 34, 117, 175).

[3] D. Purves et al., *Neuroscience*, 3rd ed, Sunderland, Mass: Sinauer Associates, Publishers, 2004, 1 p., ISBN: 978-0-87893-725-7 (cit. on pp. 1, 4, 5, 7, 19).

[4] E. R. Kandel, *Principles of Neural Science*, 5th ed, New York: McGraw-Hill, 2013, 1 p., ISBN: 978-0-07-181001-2 (cit. on pp. 1, 4–7, 19).

[5] O. Sporns, *The Complex Brain: Connectivity, Dynamics, Information*, Trends in Cognitive Sciences **26** (2022) 1066, ISSN: 13646613, (visited on 03/05/2024) (cit. on p. 1).

[6] P. Dayan and L. F. Abbott, *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*, Computational Neuroscience, Cambridge, Mass: Massachusetts Institute of Technology Press, 2001, 460 pp., ISBN: 978-0-262-04199-7 (cit. on pp. 1, 4–6, 8, 9, 11–13, 34).

[7] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition*, 1st ed., Cambridge University Press, 2014, ISBN: 978-1-107-06083-8 978-1-107-63519-7 978-1-107-44761-5, (visited on 04/24/2024) (cit. on pp. 1, 3, 6, 8, 10, 13, 19).

[8] J. Schmidhuber, *Deep Learning in Neural Networks: An Overview*, Neural Networks **61** (2015) 85, ISSN: 08936080, (visited on 12/21/2024) (cit. on p. 1).

[9] B. Zalc, "The Acquisition of Myelin: A Success Story", *Novartis Foundation Symposia*, ed. by D. J. Chadwick and J. Goode, 1st ed., vol. 276, Wiley, 2006 15, ISBN: 978-0-470-01860-6 978-0-470-03224-4, (visited on 11/23/2023) (cit. on p. 1).

[10] K. E. Garcia, C. D. Kroenke, and P. V. Bayly,
Mechanics of Cortical Folding: Stress, Growth and Stability,
Philosophical Transactions of the Royal Society B: Biological Sciences **373** (2018) 20170321,
ISSN: 0962-8436, 1471-2970, (visited on 11/23/2023) (cit. on pp. 1, 38).

[11] A. Alreja, I. Nemenman, and C. J. Rozell, *Constrained Brain Volume in an Efficient Coding Model Explains the Fraction of Excitatory and Inhibitory Neurons in Sensory Cortices*,
PLOS Computational Biology **18** (2022) e1009642, ed. by L. J. Graham, ISSN: 1553-7358,
(visited on 02/17/2022) (cit. on pp. 1, 3, 6, 36, 38).

[12] A. Dembo and T. Kailath, *Model-Free Distributed Learning*,
IEEE Transactions on Neural Networks **1** (1990) 58, ISSN: 10459227, (visited on 07/09/2024)
(cit. on pp. 1, 21, 25, 37).

[13] G. Cauwenberghs,
A Fast Stochastic Error-Descent Algorithm for Supervised Learning and Optimization,
Morgan Kaufmann, Burlington **5** (1993) 245 (cit. on pp. 1, 2, 21, 22, 25, 37).

[14] J. Werfel, X. Xie, and H. S. Seung,
Learning Curves for Stochastic Gradient Descent in Linear Feedforward Networks,
Neural Computation **17** (2005) 2699, ISSN: 0899-7667, 1530-888X, (visited on 01/19/2022)
(cit. on pp. 1, 2, 21, 22, 24, 25, 30, 37, 115).

[15] I. R. Fiete and H. S. Seung,
Gradient Learning in Spiking Neural Networks by Dynamic Perturbation of Conductances,
Physical Review Letters **97** (2006) 048104, ISSN: 0031-9007, 1079-7114,
(visited on 10/28/2019) (cit. on pp. 1, 21, 22, 37).

[16] J. T. Trachtenberg et al.,
Long-Term in Vivo Imaging of Experience-Dependent Synaptic Plasticity in Adult Cortex,
Nature **420** (2002) 788, ISSN: 0028-0836, 1476-4687, (visited on 11/20/2024) (cit. on pp. 2, 20).

[17] N. Yasumatsu, M. Matsuzaki, T. Miyazaki, J. Noguchi, and H. Kasai,
Principles of Long-Term Dynamics of Dendritic Spines,
The Journal of Neuroscience **28** (2008) 13592, ISSN: 0270-6474, 1529-2401,
(visited on 11/20/2024) (cit. on pp. 2, 20).

[18] A. Rubinski and N. E. Ziv, *Remodeling and Tenacity of Inhibitory Synapses: Relationships with Network Activity and Neighboring Excitatory Synapses*,
PLOS Computational Biology **11** (2015) e1004632, ed. by K. T. Blackwell, ISSN: 1553-7358,
(visited on 11/22/2024) (cit. on pp. 2, 20).

[19] N. E. Ziv and N. Brenner,
Synaptic Tenacity or Lack Thereof: Spontaneous Remodeling of Synapses,
Trends in Neurosciences **41** (2018) 89, ISSN: 01662236, (visited on 11/20/2024)
(cit. on pp. 2, 19, 20).

[20] U. Frey and R. G. M. Morris, *Synaptic Tagging and Long-Term Potentiation*,
Nature **385** (1997) 533, ISSN: 0028-0836, 1476-4687, (visited on 11/25/2024) (cit. on pp. 2, 20).

[21] R. L. Redondo and R. G. M. Morris,
Making Memories Last: The Synaptic Tagging and Capture Hypothesis,
Nature Reviews Neuroscience **12** (2011) 17, ISSN: 1471-003X, 1471-0048,
(visited on 11/20/2024) (cit. on pp. 2, 20, 21).

[22] M. Z. Bin Ibrahim, Z. Wang, and S. Sajikumar, *Synapses Tagged, Memories Kept: Synaptic Tagging and Capture Hypothesis in Brain Health and Disease*,
Philosophical Transactions of the Royal Society B: Biological Sciences **379** (2024) 20230237,
ISSN: 0962-8436, 1471-2970, (visited on 11/26/2024) (cit. on pp. 2, 20, 21).

[23] S Sajikumar and J. U. Frey,
Late-Associativity, Synaptic Tagging, and the Role of Dopamine during LTP and LTD,
Neurobiology of Learning and Memory **82** (2004) 12, ISSN: 10747427, (visited on 11/25/2024)
(cit. on pp. 2, 21).

[24] M. Shivarama Shetty, S. Gopinadhan, and S. Sajikumar,
Dopamine D1/D5 Receptor Signaling Regulates Synaptic Cooperation and Competition in Hippocampal CA 1 Pyramidal Neurons via Sustained ERK 1/2 Activation,
Hippocampus **26** (2016) 137, ISSN: 1050-9631, 1098-1063, (visited on 11/26/2024)
(cit. on pp. 2, 21).

[25] N. Eshel et al., *Arithmetic and Local Circuitry Underlying Dopamine Prediction Errors*,
Nature **525** (2015) 243, ISSN: 0028-0836, 1476-4687, (visited on 11/30/2024) (cit. on pp. 2, 21).

[26] O. D. Öz¸cete, A. Banerjee, and P. S. Kaeser,
Mechanisms of Neuromodulatory Volume Transmission, Molecular Psychiatry **29** (2024) 3680,
ISSN: 1359-4184, 1476-5578, (visited on 11/26/2024) (cit. on pp. 2, 21, 40).

[27] J. C. Magee and C. Grienberger, *Synaptic Plasticity Forms and Functions*,
Annual Review of Neuroscience **43** (2020) 95, ISSN: 0147-006X, 1545-4126,
(visited on 11/26/2024) (cit. on pp. 2, 21).

[28] T. Miconi, *Biologically Plausible Learning in Recurrent Neural Networks Reproduces Neural Dynamics Observed during Cognitive Tasks*, **eLife** (2017) 24 (cit. on pp. 2, 22, 25, 26, 30, 37).

[29] R. J. Williams,
Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning,
Machine Learning **8** (1992) 28 (cit. on pp. 2, 3, 22–24, 29, 37, 116).

[30] I. R. Fiete, M. S. Fee, and H. S. Seung, *Model of Birdsong Learning Based on Gradient Estimation by Dynamic Perturbation of Neural Conductances*,
Journal of Neurophysiology **98** (2007) 2038, ISSN: 0022-3077, 1522-1598,
(visited on 11/04/2024) (cit. on pp. 2, 30, 37).

[31] A. K. Dhawale, M. A. Smith, and B. P. Ölveczky, *The Role of Variability in Motor Learning*,
Annual Review of Neuroscience **40** (2017) 479, ISSN: 0147-006X, 1545-4126,
(visited on 10/22/2018) (cit. on pp. 2, 30, 37).

[32] R. H. R. Hahnloser, A. A. Kozhevnikov, and M. S. Fee,
An Ultra-Sparse Code Underliesthe Generation of Neural Sequences in a Songbird,
Nature **419** (2002) 65, ISSN: 0028-0836, 1476-4687, (visited on 01/14/2025) (cit. on pp. 2, 37).

[33] K. Doya and T. J. Sejnowski, "A Computational Model of Birdsong Learning by Auditory Experience and Auditory Feedback", *Central Auditory Processing and Neural Modeling*, ed. by P. W. F. Poon and J. F. Brugge, Boston, MA: Springer US, 1998 77, ISBN: 978-1-4613-7441-1 978-1-4615-5351-9, (visited on 11/04/2024) (cit. on p. 2).

[34] T. P. Lillicrap, D. Cownden, D. B. Tweed, and C. J. Akerman, *Random Synaptic Feedback Weights Support Error Backpropagation for Deep Learning*, Nature Communications **7** (2016) 13276, ISSN: 2041-1723, (visited on 12/18/2024) (cit. on pp. 2, 30, 37, 40).

[35] T. P. Lillicrap, A. Santoro, L. Marris, C. J. Akerman, and G. Hinton, *Backpropagation and the Brain*, Nature Reviews Neuroscience **21** (2020) 335, ISSN: 1471-003X, 1471-0048, (visited on 09/03/2020) (cit. on pp. 2, 19, 21, 30, 37).

[36] A. Payeur, J. Guerguiev, F. Zenke, B. A. Richards, and R. Naud, *Burst-Dependent Synaptic Plasticity Can Coordinate Learning in Hierarchical Circuits*, Nature Neuroscience (2021), ISSN: 1097-6256, 1546-1726, (visited on 06/23/2021) (cit. on pp. 2, 30, 37).

[37] J. P. Cunningham and B. M. Yu, *Dimensionality Reduction for Large-Scale Neural Recordings*, Nature Neuroscience **17** (2014) 1500, ISSN: 1097-6256, 1546-1726, (visited on 12/22/2024) (cit. on p. 2).

[38] M. Del Giudice, *Effective Dimensionality: A Tutorial*, Multivariate Behavioral Research **56** (2021) 527, ISSN: 0027-3171, 1532-7906, (visited on 09/22/2021) (cit. on pp. 2, 14, 15).

[39] P. Gao et al., *A Theory of Multineuronal Dimensionality, Dynamics and Measurement*, preprint, Neuroscience, 2017, (visited on 09/22/2021) (cit. on pp. 2, 14).

[40] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever, *Evolution Strategies as a Scalable Alternative to Reinforcement Learning*, 2017, arXiv: 1703.03864 [stat], (visited on 12/27/2024), pre-published (cit. on pp. 2, 39).

[41] D. H. Hubel and T. N. Wiesel, *Receptive Fields, Binocular Interaction and Functional Architecture in the Cat's Visual Cortex*, The Journal of Physiology **160** (1962) 106, ISSN: 0022-3751, 1469-7793, (visited on 02/02/2024) (cit. on pp. 2, 34, 37).

[42] I. Nauhaus, K. J. Nielsen, and E. M. Callaway, *Efficient Receptive Field Tiling in Primate V1*, Neuron **91** (2016) 893, ISSN: 08966273, (visited on 02/02/2024) (cit. on pp. 2, 34, 37).

[43] H. Ko et al., *Functional Specificity of Local Synaptic Connections in Neocortical Networks*, Nature **473** (2011) 87, ISSN: 0028-0836, 1476-4687, (visited on 09/20/2022) (cit. on pp. 2, 9, 34, 38, 39).

[44] L. Cossell et al., *Functional Organization of Excitatory Synaptic Strength in Primary Visual Cortex*, Nature **518** (2015) 399, ISSN: 0028-0836, 1476-4687, (visited on 09/20/2022) (cit. on pp. 2, 9, 34, 37–39).

[45] Z. Ding et al., *Functional Connectomics Reveals General Wiring Rule in Mouse Visual Cortex*, preprint, bioRxiv, 2023, (visited on 03/01/2024) (cit. on pp. 2, 9, 34, 38, 39).

[46] S. N. Chettih and C. D. Harvey,
Single-Neuron Perturbations Reveal Feature-Specific Competition in V1, Nature **567** (2019) 334, ISSN: 0028-0836, 1476-4687, (visited on 09/20/2022) (cit. on pp. 2, 9, 34, 38, 39).

[47] I. A. Oldenburg et al., *The Logic of Recurrent Circuits in the Primary Visual Cortex*, Nature Neuroscience **27** (2024) 137, ISSN: 1097-6256, 1546-1726, (visited on 03/05/2024) (cit. on pp. 2, 9, 34, 38, 39).

[48] M. Boerlin, C. K. Machens, and S. Denève,
Predictive Coding of Dynamical Variables in Balanced Spiking Networks,
PLoS Computational Biology **9** (2013) e1003258, ed. by O. Sporns, ISSN: 1553-7358, (visited on 10/12/2021) (cit. on pp. 3, 9–11, 17, 34, 36, 38).

[49] S. Denève and C. K. Machens, *Efficient Codes and Balanced Networks*, Nature Neuroscience **19** (2016) 375, ISSN: 1097-6256, 1546-1726, (visited on 10/08/2021) (cit. on pp. 3, 9, 16, 17, 34, 38).

[50] M. Valente et al., *Correlations Enhance the Behavioral Readout of Neural Population Activity in Association Cortex*, Nature Neuroscience **24** (2021) 975, ISSN: 1097-6256, 1546-1726, (visited on 04/20/2022) (cit. on pp. 3, 34, 38).

[51] B. K. Murphy and K. D. Miller, *Balanced Amplification: A New Mechanism of Selective Amplification of Neural Activity Patterns*, Neuron **61** (2009) 635, ISSN: 08966273, (visited on 04/23/2019) (cit. on pp. 3, 14, 16, 18, 19, 34, 35, 38).

[52] S. Lim and M. S. Goldman,
Balanced Cortical Microcircuitry for Maintaining Information in Working Memory,
Nature Neuroscience **16** (2013) 1306, ISSN: 1097-6256, 1546-1726, (visited on 12/06/2018) (cit. on pp. 3, 34, 38).

[53] D. Chklovskii and C. Stevens, "Wiring Optimization in the Brain",
Advances in Neural Information Processing Systems, ed. by S. Solla, T. Leen, and K. Müller, vol. 12, MIT Press, 1999 (cit. on pp. 3, 36, 38).

[54] J. A. Perge, J. E. Niven, E. Mugnaini, V. Balasubramanian, and P. Sterling,
Why Do Axons Differ in Caliber?, The Journal of Neuroscience **32** (2012) 626,
ISSN: 0270-6474, 1529-2401, (visited on 06/17/2024) (cit. on pp. 3, 36, 38).

[55] T. P. Lillicrap and A. Santoro, *Backpropagation through Time and the Brain*,
Current Opinion in Neurobiology **55** (2019) 82, ISSN: 09594388, (visited on 01/14/2025) (cit. on p. 3).

[56] A. Nayebi et al., *Recurrent Connections in the Primate Ventral Visual Stream Mediate a Trade-Off Between Task Performance and Network Size During Core Object Recognition*, Neural Computation **34** (2022) 1652, ISSN: 0899-7667, 1530-888X, (visited on 01/05/2023) (cit. on p. 3).

[57] M. V. Tsodyks, W. E. Skaggs, T. J. Sejnowski, and B. L. McNaughton,
Paradoxical Effects of External Modulation of Inhibitory Interneurons,
The Journal of Neuroscience **17** (1997) 4382, ISSN: 0270-6474, 1529-2401,
(visited on 01/21/2022) (cit. on p. 3).

[58] H. Ozeki, I. M. Finn, E. S. Schaffer, K. D. Miller, and D. Ferster,
Inhibitory Stabilization of the Cortical Network Underlies Visual Surround Suppression,
Neuron **62** (2009) 578, ISSN: 08966273, (visited on 12/30/2024) (cit. on pp. 3, 38).

[59] A. Sanzeni, M. H. Histed, and N. Brunel,
Response Nonlinearities in Networks of Spiking Neurons,
PLOS Computational Biology **16** (2020) e1008165, ed. by H. Cuntz, ISSN: 1553-7358,
(visited on 09/24/2024) (cit. on pp. 3, 17, 38).

[60] C. Van Vreeswijk and H. Sompolinsky,
Chaos in Neuronal Networks with Balanced Excitatory and Inhibitory Activity,
Science **274** (1996) 1724, ISSN: 0036-8075, 1095-9203, (visited on 11/09/2023)
(cit. on pp. 3, 14, 16–18).

[61] C. van Vreeswijk and H. Sompolinsky, *Chaotic Balanced State in a Model of Cortical Circuits*,
Neural Computation **10** (1998) 1321, ISSN: 0899-7667, 1530-888X, (visited on 12/20/2022)
(cit. on pp. 3, 16).

[62] Y. Ahmadian and K. D. Miller, *What Is the Dynamical Regime of Cerebral Cortex?*,
Neuron **109** (2021) 3373, ISSN: 08966273, (visited on 06/01/2022) (cit. on pp. 3, 16, 17).

[63] M. Okun and I. Lampl, *Instantaneous Correlation of Excitation and Inhibition during Ongoing and Sensory-Evoked Activities*, Nature Neuroscience **11** (2008) 535,
ISSN: 1097-6256, 1546-1726, (visited on 11/07/2023) (cit. on pp. 3, 35).

[64] M. Graupner and A. D. Reyes, *Synaptic Input Correlations Leading to Membrane Potential Decorrelation of Spontaneous Activity in Cortex*, The Journal of Neuroscience **33** (2013) 15075,
ISSN: 0270-6474, 1529-2401, (visited on 11/07/2023) (cit. on pp. 3, 35).

[65] J. Barral and A. D. Reyes, *Synaptic Scaling Rule Preserves Excitatory–Inhibitory Balance and Salient Neuronal Network Dynamics*, Nature Neuroscience **19** (2016) 1690,
ISSN: 1097-6256, 1546-1726, (visited on 11/08/2023) (cit. on p. 3).

[66] E. M. Izhikevich,
Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting,
Computational Neuroscience, Cambridge, Mass: MIT press, 2007, ISBN: 978-0-262-09043-8
(cit. on p. 3).

[67] B. Gutkin and F. Zeldenrust, *Spike Frequency Adaptation*, Scholarpedia **9** (2014) 30643
(cit. on p. 3).

[68] M. F. Bear, B. W. Connors, and M. A. Paradiso, *Neuroscience: Exploring the Brain*, 4. ed,
Philadelphia: Wolters Kluwer, 2016, 975 pp., ISBN: 978-0-7817-7817-6 978-1-4511-0954-2
(cit. on pp. 4, 6–9, 12, 19).

[69] C. S. Von Bartheld, J. Bahney, and S. Herculano-Houzel, *The Search for True Numbers of Neurons and Glial Cells in the Human Brain: A Review of 150 Years of Cell Counting*,
Journal of Comparative Neurology **524** (2016) 3865, ISSN: 0021-9967, 1096-9861,
(visited on 08/28/2024) (cit. on p. 4).

[70] H. Hering and M. Sheng,
DENDRITIC SPINES: STRUCTURE, DYNAMICS AND REGULATION, (2001) (cit. on p. 6).

[71] M. London and M. Häusser, *DENDRITIC COMPUTATION*, (2005) (cit. on p. 6).

[72] BRAIN Initiative Cell Census Network (BICCN) et al.,
A Multimodal Cell Census and Atlas of the Mammalian Primary Motor Cortex,
Nature **598** (2021) 86, ISSN: 0028-0836, 1476-4687, (visited on 10/26/2021) (cit. on pp. 6, 7).

[73] S. Genon, A. Reid, R. Langner, K. Amunts, and S. B. Eickhoff,
How to Characterize the Function of a Brain Region,
Trends in Cognitive Sciences **22** (2018) 350, ISSN: 13646613, (visited on 08/30/2024)
(cit. on p. 7).

[74] R. Brette, *Philosophy of the Spike: Rate-Based vs. Spike-Based Theories of the Brain*,
Frontiers in Systems Neuroscience **9** (2015), ISSN: 1662-5137, (visited on 05/06/2019)
(cit. on pp. 8, 12, 13, 16).

[75] I. Lampl, J. S. Anderson, D. C. Gillespie, and D. Ferster, *Prediction of Orientation Selectivity from Receptive Field Architecture in Simple Cells of Cat Visual Cortex*, Neuron **30** (2001) 263,
ISSN: 08966273, (visited on 05/02/2023) (cit. on p. 8).

[76] J. Touryan, G. Felsen, and Y. Dan,
Spatial Structure of Complex Cell Receptive Fields Measured with Natural Images,
Neuron **45** (2005) 781, ISSN: 08966273, (visited on 09/11/2024) (cit. on p. 9).

[77] R. Clay Reid and J.-M. Alonso,
Specificity of Monosynaptic Connections from Thalamus to Visual Cortex,
Nature **378** (1995) 281, ISSN: 0028-0836, 1476-4687, (visited on 02/27/2023) (cit. on pp. 9, 39).

[78] A. Smith, *Estimating Receptive Field Size from fMRI Data in Human Striate and Extrastriate Visual Cortex*, Cerebral Cortex **11** (2001) 1182, ISSN: 14602199, (visited on 02/03/2025)
(cit. on p. 9).

[79] J. Freeman and E. P. Simoncelli, *Metamers of the Ventral Stream*,
Nature Neuroscience **14** (2011) 1195, ISSN: 1097-6256, 1546-1726, (visited on 02/03/2025)
(cit. on p. 9).

[80] J. H. Siegle et al., *Survey of Spiking in the Mouse Visual System Reveals Functional Hierarchy*,
Nature **592** (2021) 86, ISSN: 0028-0836, 1476-4687, (visited on 02/05/2025) (cit. on p. 9).

[81] A. D. Lien and M. Scanziani,
Tuned Thalamic Excitation Is Amplified by Visual Cortical Circuits,
Nature Neuroscience **16** (2013) 1315, ISSN: 1097-6256, 1546-1726, (visited on 02/23/2023)
(cit. on pp. 9, 39).

[82] J. Kadmon, J. Timcheck, and S. Ganguli,
"Predictive Coding in Balanced Neural Networks with Noise, Chaos and Delays",
Advances in Neural Information Processing Systems,
ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, vol. 33,
Curran Associates, Inc., 2020 16677 (cit. on pp. 9, 34, 38).

[83] A. Renart et al., *The Asyncronous State in Cortical Circuits*, Science **327** (2010),
(visited on 09/25/2018) (cit. on pp. 12, 16, 38).

[84] A. Marin Vargas et al.,
Task-Driven Neural Network Models Predict Neural Dynamics of Proprioception,
Cell **187** (2024) 1745, ISSN: 00928674, (visited on 01/04/2025) (cit. on p. 12).

[85] M. M. Churchland, J. P. Cunningham, M. T. Kaufman, S. I. Ryu, and K. V. Shenoy, *Cortical Preparatory Activity: Representation of Movement or First Cog in a Dynamical Machine?*, Neuron **68** (2010) 387, ISSN: 08966273, (visited on 10/07/2024) (cit. on p. 12).

[86] V. Mante, D. Sussillo, K. V. Shenoy, and W. T. Newsome, *Context-Dependent Computation by Recurrent Dynamics in Prefrontal Cortex*, Nature **503** (2013) 78, ISSN: 0028-0836, 1476-4687, (visited on 04/03/2019) (cit. on p. 12).

[87] K. Wimmer, D. Q. Nykamp, C. Constantinidis, and A. Compte, *Bump Attractor Dynamics in Prefrontal Cortex Explains Behavioral Precision in Spatial Working Memory*, Nature Neuroscience **17** (2014) 431, ISSN: 1097-6256, 1546-1726, (visited on 10/07/2024) (cit. on p. 12).

[88] B. DePasquale, D. Sussillo, L. Abbott, and M. M. Churchland, *The Centrality of Population-Level Factors to Network Computation Is Demonstrated by a Versatile Approach for Training Spiking Networks*, Neuron **111** (2023) 631, ISSN: 08966273, (visited on 03/07/2023) (cit. on pp. 12–14).

[89] M. Dipoppa et al., *Vision and Locomotion Shape the Interactions between Neuron Types in Mouse Visual Cortex*, Neuron **98** (2018) 602, ISSN: 08966273, (visited on 10/07/2024) (cit. on p. 12).

[90] B. R. Cowley et al., *Mapping Model Units to Visual Neurons Reveals Population Code for Social Behaviour*, Nature **629** (2024) 1100, ISSN: 0028-0836, 1476-4687, (visited on 10/07/2024) (cit. on p. 12).

[91] J. K. Lappalainen et al., *Connectome-Constrained Networks Predict Neural Activity across the Fly Visual System*, Nature (2024), ISSN: 0028-0836, 1476-4687, (visited on 10/07/2024) (cit. on p. 12).

[92] M. W. Hirsch and S. Smale, *Differential Equations, Dynamical Systems, and Linear Algebra*, Pure and Applied Mathematics 60, San Diego [u.a.]: Acad. Press, 1974, XI, 358, ISBN: 0-12-349550-4 (cit. on p. 14).

[93] S. J. Axler, *Linear Algebra Done Right*, 3rd ed., Undergraduate Texts in Mathematics, Springer International Publishing, 2015, ISBN: 978-3-319-11080-6 3-319-11080-2 (cit. on pp. 14, 15, 18, 19).

[94] H. J. Sommers, A. Crisanti, H. Sompolinsky, and Y. Stein, *Spectrum of Large Random Asymmetric Matrices*, Physical Review Letters **60** (1988) 1895, ISSN: 0031-9007, (visited on 01/04/2025) (cit. on p. 14).

[95] T. Tao, V. Vu, and M. Krishnapur, *Random Matrices: Universality of ESDs and the Circular Law*, The Annals of Probability **38** (2010), ISSN: 0091-1798, (visited on 01/04/2025) (cit. on p. 14).

[96] C. Giraud, *Introduction to High-Dimensional Statistics*, Chapman & Hall/CRC Monographs on Statistics & Applied Probability, Taylor & Francis, 2014, ISBN: 978-1-4822-3794-8 (cit. on p. 15).

[97] M. Greenacre et al., *Principal Component Analysis*, Nature Reviews Methods Primers **2** (2022) 100, ISSN: 2662-8449 (cit. on p. 15).

[98] M. Jazayeri and S. Ostojic, *Interpreting Neural Computations by Examining Intrinsic and Embedding Dimensionality of Neural Activity*, Current Opinion in Neurobiology **70** (2021) 113, ISSN: 09594388, (visited on 10/30/2024) (cit. on p. 15).

[99] W. Softky and C Koch, *The Highly Irregular Firing of Cortical Cells Is Inconsistent with Temporal Integration of Random EPSPs*, The Journal of Neuroscience **13** (1993) 334, ISSN: 0270-6474, 1529-2401, (visited on 10/28/2024) (cit. on p. 16).

[100] G. Hennequin, E. J. Agnes, and T. P. Vogels, *Inhibitory Plasticity: Balance, Control, and Codependence*, Annual Review of Neuroscience **40** (2017) 557, ISSN: 0147-006X, 1545-4126, (visited on 02/07/2025) (cit. on pp. 16, 17, 38).

[101] N. Brunel, *Dynamics of Sparsely Connected Networks of Excitatory and Inhibitory Spiking Neurons*, Computational Neuroscience **8** (2000) 183 (cit. on p. 17).

[102] A. Sanzeni et al., *Mechanisms Underlying Reshuffling of Visual Responses by Optogenetic Stimulation in Mice and Monkeys*, Neuron **111** (2023) 4102, ISSN: 08966273, (visited on 06/25/2024) (cit. on p. 17).

[103] T. Kenet, D. Bibitchkov, M. Tsodyks, A. Grinvald, and A. Arieli, *Spontaneously Emerging Cortical Representations of Visual Attributes*, Nature **425** (2003) 954, ISSN: 0028-0836, 1476-4687, (visited on 10/31/2024) (cit. on p. 18).

[104] B. Haider, A. Duque, A. R. Hasenstaub, and D. A. McCormick, *Neocortical Network Activity* In Vivo *Is Generated through a Dynamic Balance of Excitation and Inhibition*, The Journal of Neuroscience **26** (2006) 4535, ISSN: 0270-6474, 1529-2401, (visited on 03/07/2024) (cit. on p. 18).

[105] A. Sanzeni et al., *Inhibition Stabilization Is a Widespread Property of Cortical Networks*, eLife **9** (2020) e54875, ISSN: 2050-084X, (visited on 12/30/2024) (cit. on p. 18).

[106] D. E. Rumelhart, G. E. Hintont, and R. J. Williams, *Learning Representations by Back-Propagating Errors*, (1986) (cit. on pp. 19, 21).

[107] A. Minerbi et al., *Long-Term Relationships between Synaptic Tenacity, Synaptic Remodeling, and Network Activity*, PLoS Biology **7** (2009) e1000136, ed. by C. F. Stevens, ISSN: 1545-7885, (visited on 02/23/2024) (cit. on p. 20).

[108] G.-q. Bi and M.-m. Poo, *Synaptic Modifications in Cultured Hippocampal Neurons: Dependence on Spike Timing, Synaptic Strength, and Postsynaptic Cell Type*, The Journal of Neuroscience **18** (1998) 10464, ISSN: 0270-6474, 1529-2401, (visited on 11/26/2024) (cit. on p. 20).

[109] R. Dvorkin and N. E. Ziv, *Relative Contributions of Specific Activity Histories and Spontaneous Processes to Size Remodeling of Glutamatergic Synapses*, PLOS Biology **14** (2016) e1002572, ed. by E. M. Schuman, ISSN: 1545-7885, (visited on 11/21/2024) (cit. on p. 20).

[110] S. Rumpel and J. Triesch, *The Dynamic Connectome*, e-Neuroforum **7** (2016) 48, ISSN: 1868-856X, (visited on 01/04/2025) (cit. on p. 20).

[111] A. J. Holtmaat et al., *Transient and Persistent Dendritic Spines in the Neocortex In Vivo*, Neuron **45** (2005) 279, ISSN: 08966273, (visited on 11/20/2024) (cit. on p. 20).

[112] Y. F. Kalle Kossio, S. Goedeke, C. Klos, and R.-M. Memmesheimer, *Drifting Assemblies for Persistent Memory: Neuron Transitions and Unsupervised Compensation*, Proceedings of the National Academy of Sciences **118** (2021) e2023832118, ISSN: 0027-8424, 1091-6490, (visited on 01/21/2025) (cit. on p. 20).

[113] P. Manz and R.-M. Memmesheimer, *Purely STDP-based Assembly Dynamics: Stability, Learning, Overlaps, Drift and Aging*, PLOS Computational Biology **19** (2023) e1011006, ed. by T. Serre, ISSN: 1553-7358, (visited on 11/27/2024) (cit. on p. 20).

[114] N. Frémaux and W. Gerstner, *Neuromodulated Spike-Timing-Dependent Plasticity, and Theory of Three-Factor Learning Rules*, Frontiers in Neural Circuits **9** (2016), ISSN: 1662-5110, (visited on 11/26/2024) (cit. on pp. 21, 22).

[115] M. A. Farries and A. L. Fairhall, *Reinforcement Learning With Modulated Spike Timing–Dependent Synaptic Plasticity*, Journal of Neurophysiology **98** (2007) 3648, ISSN: 0022-3077, 1522-1598, (visited on 11/20/2024) (cit. on p. 21).

[116] E. M. Izhikevich, *Solving the Distal Reward Problem through Linkage of STDP and Dopamine Signaling*, Cerebral Cortex **17** (2007) 2443, ISSN: 1047-3211, 1460-2199, (visited on 11/20/2024) (cit. on p. 21).

[117] R. Legenstein, S. M. Chase, A. B. Schwartz, and W. Maass, *A Reward-Modulated Hebbian Learning Rule Can Explain Experimentally Observed Network Reorganization in a Brain Control Task*, Journal of Neuroscience **30** (2010) 8400, ISSN: 0270-6474, 1529-2401, (visited on 11/04/2024) (cit. on pp. 26, 40).

[118] L. Isserlis, *On a Formula for the Product-Moment Coefficient of Any Order of a Normal Frequency Distribution in Any Number of Variables*, Biometrika **12** (1918) 134, JSTOR: 2331932 (cit. on p. 27).

[119] R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth, *On the LambertW Function*, Advances in Computational Mathematics **5** (1996) 329, ISSN: 1019-7168, 1572-9044, (visited on 03/13/2024) (cit. on p. 27).

[120] P. Züge, C. Klos, and R.-M. Memmesheimer, *Supplementary Material to 'Weight vs. Node Perturbation Learning in Temporally Extended Tasks: Weight Perturbation Often Performs Similarly or Better'*, Physical Review X **13** (2023) 021006, ISSN: 2160-3308, © 2023 American Physical Society (cit. on pp. 29, 30, 72, 116).

[121] T. Teşileanu, B. Ölveczky, and V. Balasubramanian, *Rules and Mechanisms for Efficient Two-Stage Learning in Neural Circuits*, eLife **6** (2017) e20944, ISSN: 2050-084X, (visited on 12/18/2024) (cit. on pp. 30, 37).

[122] L. F. Abbott, B. DePasquale, and R.-M. Memmesheimer, *Building Functional Networks of Spiking Model Neurons*, Nature Neuroscience **19** (2016) 350, ISSN: 1097-6256, 1546-1726, (visited on 09/04/2018) (cit. on pp. 34, 38).

[123] E. Izhikevich, *Simple Model of Spiking Neurons*, IEEE Transactions on Neural Networks **14** (2003) 1569, ISSN: 1045-9227, (visited on 12/13/2021) (cit. on p. 35).

[124] M. Rigotti et al., *The Importance of Mixed Selectivity in Complex Cognitive Tasks*, Nature **497** (2013) 585, ISSN: 0028-0836, 1476-4687, (visited on 01/04/2023) (cit. on p. 35).

[125] D. Levenstein et al., *On the Role of Theory and Modeling in Neuroscience*, The Journal of Neuroscience **43** (2023) 1074, ISSN: 0270-6474, 1529-2401, (visited on 12/29/2024) (cit. on p. 36).

[126] S. Dalm, M. van Gerven, and N. Ahmad, *Effective Learning with Node Perturbation in Multi-Layer Neural Networks*, 2024, arXiv: 2310.00965 [cs], (visited on 12/29/2024), pre-published (cit. on pp. 37, 38).

[127] J. G. Fernandez, S. Keemink, and M. van Gerven, *Gradient-Free Training of Recurrent Neural Networks Using Random Perturbations*, Frontiers in Neuroscience **18** (2024) 1439155, ISSN: 1662-453X, arXiv: 2405.08967 [cs], (visited on 12/29/2024) (cit. on p. 38).

[128] N. Ahmad, E. Schrader, and M. van Gerven, *Constrained Parameter Inference as a Principle for Learning*, 2023, arXiv: 2203.13203 [cs], (visited on 12/29/2024), pre-published (cit. on p. 38).

[129] N. Hiratani, T. P. Lillicrap, Y. Mehta, and P. E. Latham, "On the Stability and Scalability of Node Perturbation Learning", *Advances in Neural Information Processing Systems*, vol. 35, Curran Associates, Inc., 2022 31929 (cit. on p. 38).

[130] M. Kaiser and C. C. Hilgetag, *Nonoptimal Component Placement, but Short Processing Paths, Due to Long-Distance Projections in Neural Systems*, PLOS Computational Biology **2** (2006) 1 (cit. on p. 38).

[131] R. F. Betzel and D. S. Bassett, *Specificity and Robustness of Long-Distance Connections in Weighted, Interareal Connectomes*, Proceedings of the National Academy of Sciences **115** (2018), ISSN: 0027-8424, 1091-6490, (visited on 05/09/2024) (cit. on p. 38).

[132] M. Huang, *Wiring Cost Minimization: A Dominant Factor in the Evolution of Brain Networks across Five Species*, () (cit. on p. 38).

[133] L. Kushnir and S. Fusi, *Neural Classifiers with Limited Connectivity and Recurrent Readouts*, The Journal of Neuroscience **38** (2018) 9900, ISSN: 0270-6474, 1529-2401, (visited on 01/25/2023) (cit. on p. 38).

[134] J. Gustafsson, J. L. Robinson, H. Zetterberg, and J. Nielsen, *Brain Energy Metabolism Is Optimized to Minimize the Cost of Enzyme Synthesis and Transport*, Proceedings of the National Academy of Sciences **121** (2024) e2305035121, ISSN: 0027-8424, 1091-6490, (visited on 12/20/2024) (cit. on p. 38).

[135] F. Pouille, A. Marin-Burgin, H. Adesnik, B. V. Atallah, and M. Scanziani, *Input Normalization by Global Feedforward Inhibition Expands Cortical Dynamic Range*, Nature Neuroscience **12** (2009) 1577, ISSN: 1097-6256, 1546-1726, (visited on 02/10/2025) (cit. on p. 38).

[136] O. Schwartz and E. P. Simoncelli, *Natural Signal Statistics and Sensory Gain Control*, Nature Neuroscience **4** (2001) 819, ISSN: 1097-6256, 1546-1726, (visited on 02/07/2025) (cit. on p. 38).

[137] M. Koyama and A. Pujala, *Mutual Inhibition of Lateral Inhibition: A Network Motif for an Elementary Computation in the Brain*, Current Opinion in Neurobiology **49** (2018) 69, ISSN: 09594388, (visited on 02/10/2025) (cit. on p. 38).

[138] A. Angelucci and K. Sainsbury, *Contribution of Feedforward Thalamic Afferents and Corticogeniculate Feedback to the Spatial Summation Area of Macaque V1 and LGN*, Journal of Comparative Neurology **498** (2006) 330, ISSN: 0021-9967, 1096-9861, (visited on 10/30/2023) (cit. on p. 39).

[139] L. Chariker, R. Shapley, and L.-S. Young, *Orientation Selectivity from Very Sparse LGN Inputs in a Comprehensive Model of Macaque V1 Cortex*, The Journal of Neuroscience **36** (2016) 12368, ISSN: 0270-6474, 1529-2401, (visited on 11/30/2022) (cit. on p. 39).

[140] D. L. Ringach, *Sparse Thalamocortical Convergence*, Current Biology **31** (2021) 2199, ISSN: 09609822, (visited on 01/30/2024) (cit. on p. 39).

[141] D. Ferster, S. Chung, and H. Wheat, *Orientation Selectivity of Thalamic Input to Simple Cells of Cat Visual Cortex*, Nature **380** (1996) 249, ISSN: 0028-0836, 1476-4687, (visited on 05/02/2023) (cit. on p. 39).

[142] J. Park et al., *Contribution of Apical and Basal Dendrites to Orientation Encoding in Mouse V1 L2/3 Pyramidal Neurons*, Nature Communications **10** (2019) 5372, ISSN: 2041-1723, (visited on 12/21/2022) (cit. on p. 39).

[143] C Anderson, M. Von Keyserlingk, L. Lidfors, and D. Weary, *Anticipatory Behaviour in Animals: A Critical Review*, Animal Welfare **29** (2020) 231, ISSN: 0962-7286, 2054-1538, (visited on 01/22/2025) (cit. on p. 39).

[144] V. V. Pulikkottil, B. P. Somashekar, and U. S. Bhalla, *Computation, Wiring, and Plasticity in Synaptic Clusters*, Current Opinion in Neurobiology **70** (2021) 101, ISSN: 09594388, (visited on 12/30/2024) (cit. on p. 40).

[145] E. J. Agnes and T. P. Vogels, *Co-Dependent Excitatory and Inhibitory Plasticity Accounts for Quick, Stable and Long-Lasting Memories in Biological Networks*, Nature Neuroscience **27** (2024) 964, ISSN: 1097-6256, 1546-1726, (visited on 12/30/2024) (cit. on p. 40).

[146] T. Kahnt and G. Schoenbaum, *The Curious Case of Dopaminergic Prediction Errors and Learning Associative Information beyond Value*, Nature Reviews Neuroscience (2025), ISSN: 1471-003X, 1471-0048, (visited on 01/22/2025) (cit. on p. 40).

[147] C. Liu, P. Goel, and P. S. Kaeser, *Spatial and Temporal Scales of Dopamine Transmission*, Nature Reviews Neuroscience **22** (2021) 345, ISSN: 1471-003X, 1471-0048, (visited on 01/23/2025) (cit. on p. 40).

[148] M. Ren, S. Kornblith, R. Liao, and G. Hinton, *Scaling Forward Gradient With Local Losses*, 2023, arXiv: 2210.03310 [cs], (visited on 04/25/2023), pre-published (cit. on p. 40).

Assistance and resources

Throughout my doctoral studies, I received supervision by Raoul-Martin Memmesheimer; at the start of my studies also from Aditya Gilra. Christian Klos and Simon Essink proofread large parts of this thesis. I collaborated with Christian Klos and Raoul-Martin Memmesheimer on publication 1 [1], as stated in the contribution statement in Chapter 3. On publication 2 [2] I collaborated with Raoul-Martin Memmesheimer, as stated in the contribution statement in Chapter 4. In addition, I profited from countless discussions in our group Neural Network Dynamics and Computation, both formal and informal.

The programming and writing for this thesis were performed on my laptop, as well as on a personal computer and the computing cluster of our group. Through the University of Bonn, I had access to various scientific journals. This thesis uses the LaTeX thesis template from the Physikalisches Institut of the University of Bonn. For the writing I used a spell checker. I used GitHub Copilot for part of the programming, but did not use AI for the writing.

I used further tools that are common and expected, like a keyboard, stationary, blackboards, etc., which are not worth mentioning.

List of Figures