

# Weight transport through spike timing for robust local gradients

Timo Gierlich<sup>\*1,2</sup>, Andreas Baumbach<sup>2</sup>, Akos F. Kungl<sup>2</sup>, Kevin Max<sup>1,3</sup>, and Mihai A. Petrovici<sup>†1</sup>

<sup>1</sup>Department of Physiology, Bern University, Switzerland

<sup>2</sup>Kirchhoff-Institut für Physik, Ruprecht-Karls-Universität Heidelberg, Germany

<sup>3</sup>Neural Computation Unit, Okinawa Institute of Science and Technology, Japan

## Abstract

In both machine learning and in computational neuroscience, plasticity in functional neural networks is frequently expressed as gradient descent on a cost. Often, this imposes symmetry constraints that are difficult to reconcile with local computation, as is required for biological networks or efficient neuromorphic hardware. For example, wake-sleep learning in networks characterized by Boltzmann distributions inherently builds on the assumption of symmetric connectivity. Similarly, the error backpropagation algorithm is notoriously plagued by the weight transport problem between the representation and the error stream. Existing solutions such as feedback alignment tend to circumvent the problem by deferring to the robustness of these algorithms to weight asymmetry. However, such solutions are known to scale poorly with network size and depth and require additional mechanisms to improve their functionality.

We introduce a complementary learning rule for spiking neural networks, which uses spike timing statistics to extract and correct the asymmetry between effective reciprocal connections. Apart from being quintessentially spike-based and fully local, our proposed mechanism takes advantage of a ubiquitous feature of physical neuronal networks: noise. Based on an interplay between Hebbian and anti-Hebbian plasticity, synapses can thereby recover the true local gradient. This also alleviates discrepancies that arise from neuron and synapse variability – an omnipresent property of physical neuronal networks, both biological and artificial. We demonstrate the efficacy of our mechanism using different spiking network models. First, we show how a combination of Hebbian and anti-Hebbian plasticity can significantly improve convergence to the target distribution in probabilistic spiking networks as compared to Hebbian plasticity alone. Second, in neuronal hierarchies based on cortical microcircuits, we show how our proposed mechanism effectively enables the alignment of feedback weights to the forward pathway, thus allowing the backpropagation of correct feedback errors.

## 1 Introduction

Prominent models of neuronal computation rely on core assumptions that inevitably give rise to symmetry constraints on their connectivity. For example, prominent re-

current network models such as Hopfield networks [1] and Boltzmann machines [2] require a symmetric weight matrix, which also needs to be enforced during wake-sleep learning [3]. They effectively inherit this property from their spin-glass archetypes in solid-state physics, for which the symmetry of particle interactions follows from fundamental laws of nature.

Perhaps even more prominently, the reverse calculation of gradients in deep neural networks naturally requires knowledge of the forward weights [4–6]. While this weight transport is inconsequential when calculations are simply carried out by an arithmetic logic unit, models of error backpropagation in the brain require the corresponding backward transport circuitry to mirror the forward one [7–9].

In general, network models for physical neuronal substrates<sup>1</sup> such as the brain or analog neuromorphic hardware are bound to constraints that are inherent to their physics. One such restriction is (spatiotemporal) locality, a fundamental property of physical networks which strongly limits the information that may enter synaptic plasticity rules. Consequently, the physical plausibility of complex, non-local learning algorithms is determined by their amenability to implementation using only local operations.

Another characteristic feature of physical neuronal computing is the inevitable presence of inherent temporal and spatial parameter variations across neurons and synapses. In the brain, these naturally emerge, for instance, from the morphological variety among different neurons of the same cell type; in analog neuromorphic hardware, component variability inevitably occurs during the manufacturing process. This expresses the need for homeostatic mechanisms additional to functional learning rules to increase robustness and maintain operational stability [10–15]. A direct and often ignored consequence of substrate heterogeneity is that the effect of the weight on a postsynaptic neuron crucially depends on the neuron parameters (synapse model, conductance values, etc.), meaning that identical weight values elicit different post-synaptic potentials (PSPs) in different neurons. What counts on the computational level however is the *effective weight*.

Hence, for promoting functionality and increasing the biological plausibility, learning algorithms for physical neuronal networks should address both the weight transport problem and the resilience to inherent parameter variability. In the

<sup>1</sup>To make the distinction between artificial neural networks (ANNs) as used in deep learning and physical, time-continuous network models for biological or bio-inspired applications, we will refer to the former type as *neural* networks and to the latter as *neuronal*.

\*timo.gierlich@unibe.ch

†mihai.petrovici@unibe.ch

following, we propose a solution to these problems in spiking networks and investigate its effectiveness for two common learning schemes that are notoriously plagued by the weight transport problem: wake-sleep learning [3] and error backpropagation (BP) [4, 6].

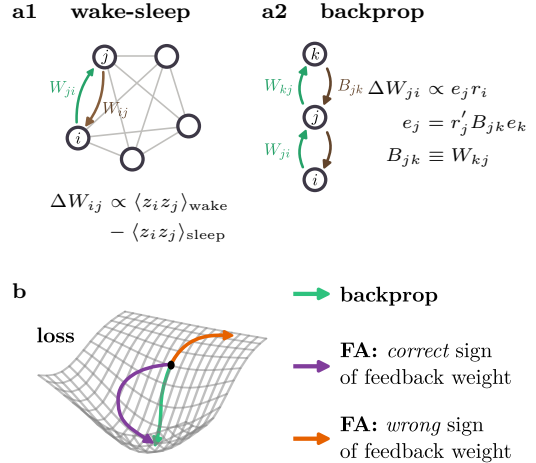
By describing spiking dynamics as sampling from an underlying Boltzmann distribution, spiking sampling networks (SSNs) create a direct link between spiking neuronal networks and Boltzmann machines. They are thus able to learn probabilistic internal representations of the world, which offers an algorithmic interpretation of certain activity patterns in the brain [16–18], while also enabling the instantiation of Bayesian generative and discriminative models in neuromorphic hardware [19–22]. However, they also inherit the weight symmetry constraints from their ANN counterparts.

Additionally, these networks are usually trained with a spiking variant of the contrastive divergence / wake-sleep algorithm [3, 23], which minimizes the difference of correlations between data-constrained and free sampling phases. Since correlations are symmetric under the exchange of neurons, weight updates of reciprocal synapses must also be identical (as would also follow directly from the general symmetry constraint on the weight matrix). However, since in physical networks the calculation of these weight updates happens in different synapses that cannot communicate directly, symmetry is not generally guaranteed. Therefore, the actual weight updates may differ significantly from the ones assumed by the algorithm; apart from deviating from the intended trajectory of learning, this may also lead to a forgetting of previously learned features. Altogether, this can result in a significant drop in performance, as we also show later.

Similar issues have initially led to a strong pushback against BP-like learning in the brain [9, 24]. With the emergence of biologically plausible adaptations of BP [25–37], several issues of standard BP have been mitigated; however, many of these algorithms still (at least implicitly) rely on copying the weights from the bottom-up pathways to the top-down pathways for correct transportation of errors; for example, approaches such as the Kolen-Pollack algorithm and variants [31–34] defer the weight transport problem to a *weight update* transport problem. Another common way to circumvent the weight transport problem is to ignore it altogether. Feedback alignment (FA) [38] builds on the observation that during training, forward weights tend to align to random, but fixed backward connections, and can therefore transport meaningful errors across layers. However, FA is known to scale poorly in deeper networks [39–41] (see also fig. 1b).

Furthermore, only few bio-plausible adaptations of BP consider networks of spiking neurons, e.g. [32, 42]. For biological realism, known solutions to weight transport such as phaseless alignment learning (PAL) [41] need to be re-evaluated for their applicability in spiking networks. In this work, we provide a fully spike-based solution to the weight transport problem, and also introduce a bio-plausible, spiking implementation of error BP based on dendritic cortical microcircuits [29, 35].

In summary, physically plausible realizations of functionally powerful learning rules require additional homeostatic mechanisms to establish robustness against inevitable parameter variability across neurons and synapses and to maintain operational stability. In this work, we address



**Figure 1: The weight transport problem in physical neuronal networks.** Many successful learning rules presuppose certain symmetry properties of the underlying network structure. This poses challenges to reconcile them with the locality principle of physical neuronal computation. **a1)** In wake-sleep learning, the network is trained using correlation measurements. Since these measurements are carried out by two distinct, individually parameterized synapses (here  $W_{ji}$  and  $W_{ij}$ ), weight updates are not symmetric. **a2)** The BP algorithm relies on the transportation of the gradient across layers, which requires a copy of the forward weights (green) to the backward path (blue). **b)** Depending on the misalignment between the true BP gradient (green arrow) and the trajectory followed by FA, learning can be slowed down significantly (violet) or fail completely (orange).

these problems and propose that nature has found a solution to the challenges of weight transport and robustness in noisy inhomogeneous substrates. We demonstrate that a particular form of spike-timing-dependent plasticity (STDP) is capable of aligning the synaptic weights of reciprocally connected neurons in a recurrently connected network. We therefore call this framework spike-based alignment learning (SAL). Similarly to its rate-based sibling PAL [41], SAL is designed to augment existing spike-based learning rules that are subject to the weight transport problem.

To this end, we combine three essential, but otherwise basic elements of cortical dynamics. We leverage the *spike-based communication* between neurons in order to directly access temporal discrepancies arising from asymmetries in the effective connectivity. However, these only become tractable in the presence of *noise*, here an explicit feature rather than a bug. Ultimately, the resulting information can enter a modified *STDP rule* which thereby becomes capable to correct undesirable deviations.

## 2 Results

### 2.1 Neuron and synapse model

Consider two neurons  $i$  and  $j$  that are mutually connected through the weights  $W_{ij}$  and  $W_{ji}$  and are affected by different types of spatial and temporal noise (see fig. 2a). On analog neuromorphic chips, hardware components are subject to inevitable variability in the manufacturing process, the so-called fixed pattern noise, that cannot be fully compensated by calibration [43–45]. Likewise, neurons of the same type vary in their morphology such as cell body size and dendritic tree structure, as well as physiological prop-

erties such as ion channel density [46, 47]. These factors critically influence the input-output relationship of different neurons [48, 49]. Hence, what counts from a computational point of view is not so much a parameter of the synapse alone, but rather the *effective weight*, i.e., the effect of a synaptic event on the postsynaptic firing probability, which is what we describe as  $W$ .

Temporal noise is also omnipresent in any physical substrate. The list of sources include thermal noise, sensory noise or the stochastic nature ion channels and receptors [50]. Furthermore, cortical neurons are known to undergo constant bombardment with irregular spike trains, some of which may be considered background noise [51–54]. The situation in neuromorphic hardware is similar, with noise on neuronal membranes being either intrinsic (for analog neurons) or extrinsic, through background synaptic bombardment. We thus consider neurons to operate in a regime of stochastic spiking.

For mathematical tractability, we thus choose a generalized linear model (GLM) as our neuron model, although the working principle of SAL is of a more general nature and thus applicable to other models as well. Each neuron is described by its membrane potential  $u_i$ ,

$$u_i(t) = b_i + \sum_k W_{ik}(t) \int_0^\infty \kappa(s) S_k(t-s) ds, \quad (1)$$

where  $b_i$  is a constant bias,  $W_{ik}$  are the input weights,  $\kappa(t)$  the kernel of the PSP and  $S_k(t)$  the input spike train from neuron  $k$  (for more details see section 4.1).

The noise sources and resulting voltage fluctuations are not modeled explicitly, instead we capture the emerging randomness in the inherent probabilistic spiking mechanism of the GLM. For simplicity, we assume that the noise process has a constant mean over time, which simply enters the neuronal bias  $b_i$ , along with other constant biases such as the leak potential. The output spikes are produced by an inhomogeneous Poisson process with an absolute refractory period of length  $\tau_{\text{ref}}$ , where the instantaneous firing probability of a non-refractory neuron is given by  $r_i(t) = \tau_{\text{ref}}^{-1} \exp(u_i(t))$ .

## 2.2 Spike timing correlations reflect weight asymmetry

The working principle of SAL resides on the observation that weights leave a characteristic imprint on the cross-correlation between pre- and postsynaptic spike trains. Each spike generated by one of the neurons elicits a PSP in the respective postsynaptic partner. For an excitatory synapse, this transiently raises the postsynaptic neuron’s firing probability. The magnitude of the change is dependent on the synaptic weight, meaning that the postsynaptic neuron reacts on average more often, and especially earlier, to an incoming spike for a larger weight. Correspondingly, a spike transmitted through an inhibitory synapse would create a negative PSP, causing the postsynaptic neuron to fire less often and later. Hence, the distribution of the spike timing difference  $\Delta t_{ij} = t_j - t_i$  carries information about the synaptic weights. In the following, we will call this the spike-timing difference distribution (STDD)  $p_{ij}(\Delta t)$ . Only nearest-neighbor pre-post spike pairs are taken into account, since the timing of the first postsynaptic spike carries all relevant information.

To illustrate the effects of weight asymmetry on the STDD, consider the case of two neurons reciprocally connected by excitatory weights with  $W_{ji} > W_{ij} > 0$ . For clarity, the curves in fig. 2 are calculated for rectangular PSPs, and a detailed discussion of other kernel shapes is provided in section 2.5.

Every time neuron  $i$  spikes, neuron  $j$  will “answer” with a certain probability with a postsynaptic spike after some  $\Delta t$ , just as neuron  $j$  will respond to a spike from  $i$  fig. 2b. However, since  $W_{ji}$  is greater than  $W_{ij}$ , neuron  $j$  will respond to neuron  $i$  on average faster than vice versa. Therefore, the probability density is shifted on the right-hand side of the STDD (causal from the point of view of  $W_{ji}$ ,  $\Delta t_{ji} > 0$ ) towards the center, and on the left-hand side (anti-causal ones,  $\Delta t_{ji} < 0$ ) away from the center to the left (fig. 2c1, top panel). This leads to an asymmetry between the left and right side of the STDD, which is informative of the weight difference. Importantly, both synapses observe the same spikes and therefore the STDD of  $W_{ij}$  is the same as that of  $W_{ji}$  but mirrored at  $\Delta t = 0$ . Thereby, weight information of both synapses is directly available at each synapse and can be used for a local STDP-based plasticity rule.

The effect of excitatory reciprocal PSPs on the STDD can thus be summarized as follows. First, the higher the average value of the weights is, the more probability mass is concentrated in the interval between  $-\tau_{\text{ref}}$  and  $\tau_{\text{ref}}$ , and correspondingly less mass is contained in the tails of the distribution. Second, the greater the difference between the two weights, the more mass is transported towards  $\Delta t = 0$  on the causal half of the distribution and away from it on the anti-causal half (from the perspective of the weight that is too strong), which increases the asymmetry around  $\Delta t = 0$ .

The same principles hold if one or both synapses are inhibitory, with the only difference being that a strong inhibitory synapse pushes the probability of a post-synaptic spike to larger  $\Delta t$ . In general, we can thus conclude the following. First, the smaller the average of the two weights, the more mass is located in the tails of the distribution. Second, the greater the difference between two weights, the more mass is transported away the y-axis on the causal half of the distribution and towards it on the anti-causal half.

The key benefit of having noise in the system becomes apparent here. Without noise, single synapses might be too weak to contribute significantly to the STDD, while strong synapses would pull its entire mass towards zero. Noise allows a smooth sampling of all possible  $\Delta t$ , and allows all synapses to contribute to it.

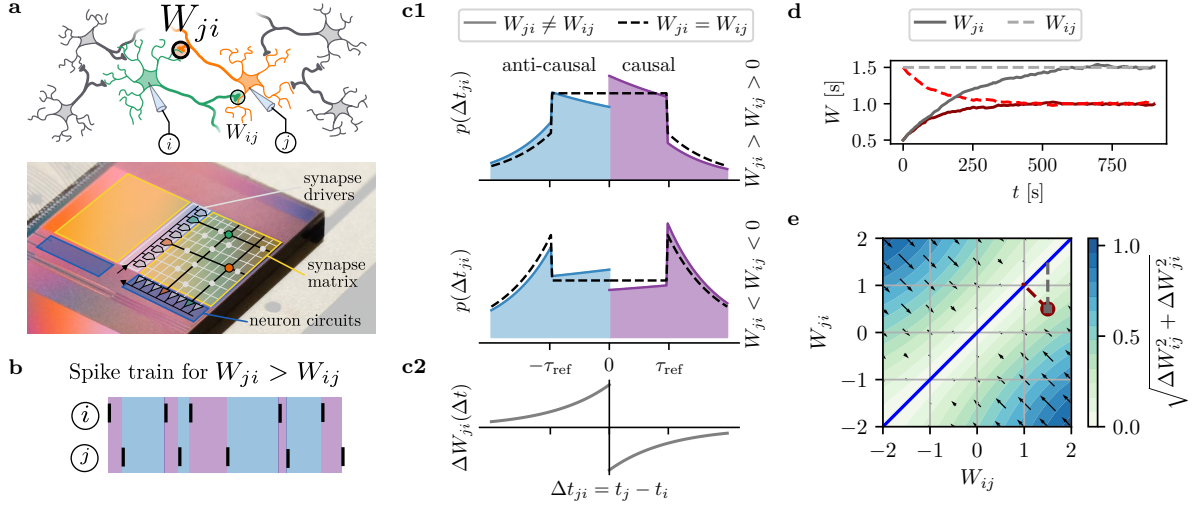
## 2.3 Spike-based alignment learning

As synapses have direct, local access to the STDD, they can use it to correct asymmetries towards their otherwise inaccessible reciprocal counterparts. Just like classical STDP harnesses the STDD for Hebbian learning, SAL uses it for symmetrization.

Instead of a classical Hebbian STDP window with a positive causal and a negative anti-causal branch, SAL uses an anti-Hebbian window (fig. 2c2):

$$\Delta W_{ij} = \begin{cases} \eta \alpha_c f^+(\Delta t_{ij}) & \text{if } \Delta t_{ij} \geq 0 \text{ (causal),} \\ \eta \alpha_a f^-(\Delta t_{ij}) & \text{if } \Delta t_{ij} < 0 \text{ (anti-causal),} \end{cases} \quad (2)$$

Here,  $\eta$  is the learning rate and  $f(\Delta t)$  describes the shape of the STDP kernel, which we assume to be  $f^\pm(\Delta t) =$



**Figure 2: Principles of spike-based alignment learning.** **a)** Two reciprocally connected neurons  $i$  and  $j$  embedded in a recurrent neural network fire stochastically due to external spike noise. Here, weight  $W_{ji}$  (orange synapse) is stronger than  $W_{ij}$  (green). Because SAL symmetrizes *effective* weights rather than pure synaptic strengths, effects of morphological differences between the two neurons or the synaptic location at the dendritic tree are implicitly balanced out. This underpins the feasibility of SAL in both biological and neuromorphic substrates (illustration adapted from [55]). **b)** Example spike trains with colored spike-timing differences  $\Delta t$  (causal in purple and anti-causal in blue), as seen from the perspective of  $W_{ji}$ . **c1)** Spike-timing difference distribution (STDD) distribution for two reciprocally connected neurons as described in section 2.2. The upper panel shows two excitatory weights, the lower two inhibitory ones. **c2)** STDP with an anti-Hebbian window used by SAL for weight alignment. **d)** Time course of symmetrization with SAL. If both weights are plastic, they converge to their common mean (red); if only one weight uses SAL, it converges to the other one (gray). **e)** Phase diagram showing the evolution of the two weights under SAL. The arrows indicate the direction of the weight update through SAL, the color map in the background the magnitude of the update. The blue line indicates the attractor of SAL which lies on the diagonal  $W_{ij} = W_{ji}$ . SAL always converges to the desired solution  $W_{ij} = W_{ji}$  from any starting point in the  $W_{ij}$ - $W_{ji}$ -plane. The example trajectories from d) are depicted in red and gray.

$\exp(\mp \Delta t / \tau_{\text{ref}})$  unless stated otherwise. The prefactors  $\alpha_{a/c}$  determine the type of plasticity, such as Hebbian, anti-Hebbian or SAL. In SAL, they are chosen such that causal spike pairs weaken the synapse ( $\alpha_c = -1$ ) and anti-causal ones strengthen it ( $\alpha_a = 1$ ). This simple plasticity rule is capable of extracting the asymmetry from the STDD to produce average weight updates that align the two weights such that they converge to their mean. Figure 2d shows the result of a numerical simulation for a pair of reciprocally connected neurons; a detailed analytical proof of the stability of fixed points under SAL is given in section 4.2.2.

The shape of the STDP window  $f$  plays a key role in extracting the information relevant for symmetrization from the STDD: For instance, an exponential window  $f(\Delta t) = \exp(\Delta t / \tau)$  “looks” primarily at  $\Delta t$ -values close to zero, where the weight differences manifest themselves as asymmetry around  $\Delta t = 0$ . The STDP time constant should roughly match the width of the PSP-kernel, because it determines the typical size of features in the STDD.

Importantly, SAL can be applied asymmetrically to reciprocal synapses. For instance, it is sufficient for only one synapse to be equipped with a SAL rule in order for it to follow its reciprocal weight (fig. 2d,e, gray). This is particularly useful in setups with distinct forward and backward streams, such as for BP, where the forward weights are learned with an error-correcting learning rule and the backward weights use SAL for alignment (see section 2.4.2). As shown in the phase plane diagram fig. 2e, SAL is able to symmetrize weights over a large parameter range.

Conceptually, the interplay between functional plasticity and SAL can be understood as follows. Under theoretically ideal circumstances (no noise, perfect initialization), net-

works that require weight symmetry would start out with reciprocal synaptic weights already lying on the diagonal of the phase plane diagram, and functional plasticity would move these weights along the diagonal. In realistic scenarios, reciprocal weights would be more randomly distributed, and functional plasticity would, in general, not drive them towards symmetry, and maybe even away from it. With SAL, plasticity receives a persistent, orthogonal drive towards its stable manifold on the diagonal, enabling the functional component to operate correctly along its orientation.

Importantly, just like classical Hebbian STDP, SAL only uses information available at the locus of the synapse, and a plasticity kernel compatible with observations from human cortex [56]. This makes it a suitable candidate for how nature might have addressed the weight transport problem. The learning rule is also compatible with implementations of STDP on neuromorphic platforms [44, 57–60], which alleviates the problem of fixed-pattern noise in analog systems and avoids expensive copy operations in digital ones.

## 2.4 SAL in functional spiking networks

SAL is designed to enhance functional learning rules that by definition require weight symmetry and are therefore difficult to reconcile with the noisy reality of physical neuronal substrates, whether biological or artificial. SAL can be implemented either in a phaseless manner or by adding reoccurring symmetrization phases during learning; in programmable hardware, such phases are easily introduced, while in biology, they can occur while the brain is not attending to external sensory stimuli, most prominently during sleep.

In the following, we present two scenarios that demonstrate the effectiveness of SAL: First, we turn to SSNs [61, 62], which require both initial weight symmetry and symmetric weight updates for exact gradient descent. We show that SSNs are sensitive to both noise on the initial weight matrix and on the STDP mechanism and how SAL improves learning under such inhomogeneous conditions.

Second, we demonstrate the efficacy of SAL in a micro-circuit model of hierarchical cortical computation [29, 35]. There, the local error correction relies on the backward transportation of gradients through cortical microcircuits, which inherits the weight transport problem from classical error backpropagation. We show that the microcircuits equipped with SAL evolve much more similarly to BP compared to FA, which is prone to misalignment.

#### 2.4.1 SAL in spiking sampling networks

In this section, we demonstrate the effectiveness of SAL in the framework of SSNs [61, 62]. As a model for the Bayesian brain, SSNs can explain characteristics of perception in noisy, ambiguous environments [63, 64]. These networks form probabilistic latent representations of the world and explore these learned internal states through spontaneous activity (sampling). Interestingly, the sampling process is driven by noise, the same resource that SAL uses for synaptic symmetrization. Due to the complex correlations that they need to learn between their constituent neurons, SSNs provide a powerful test case for SAL.

SSNs represent a natural way to mathematically formalize the dynamics of sampling in a biologically plausible manner by extending the theory of Boltzmann machines to spiking networks. However, they also inherit the need for a symmetric weight matrix from their machine learning counterparts. An SSN consists of  $N$  bidirectionally connected neurons, each modeled as a GLM following eq. (1), with absolute refractory time  $\tau_{\text{ref}}$ , rectangular PSPs, and a logistic activation function. Each neuron  $k$  is assigned a binary state:  $z_k = 1$  if the neuron is refractory and  $z_k = 0$  otherwise, resulting in a network state vector  $\mathbf{z} \in [0, 1]^N$  (see fig. 3). The probability for the network to be in a state  $\mathbf{z}$  is given by

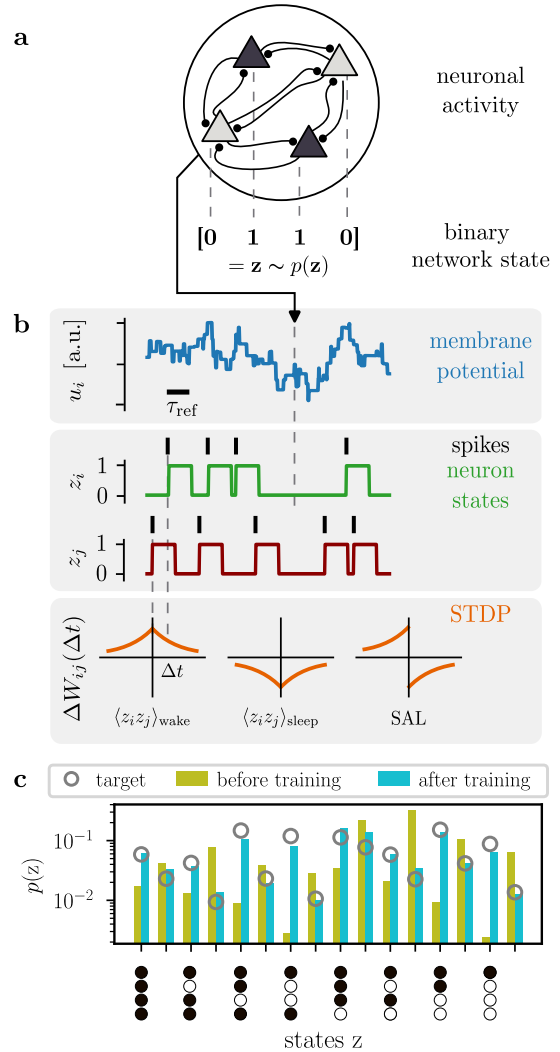
$$p(\mathbf{z}) = \frac{1}{Z} \exp \left[ \frac{1}{2} \mathbf{z}^T \mathbf{W} \mathbf{z} + \mathbf{z}^T \mathbf{b} \right], \quad (3)$$

where  $Z = \sum_{\mathbf{z}} \exp \left[ \frac{1}{2} \mathbf{z}^T \mathbf{W} \mathbf{z} + \mathbf{z}^T \mathbf{b} \right]$  represents the normalization factor,  $\mathbf{W} \in \mathbb{R}^{N \times N}$  is a symmetric weight matrix (i.e.,  $\mathbf{W} = \mathbf{W}^T$ ) and  $\mathbf{b} \in \mathbb{R}^N$  a vector containing all neuronal biases.

The network can be trained to a suitable target distribution  $p^*(\mathbf{z})$  by gradient descent on the Kullback-Leibler divergence between  $p$  and  $p^*$ . This ultimately yields the wake-sleep algorithm with a local Hebbian learning rule. Training consists of two alternating phases: a wake phase, which is constrained by the target distribution, and a sleep phase, in which the network samples freely from its current internal distribution. A weight update is then given by

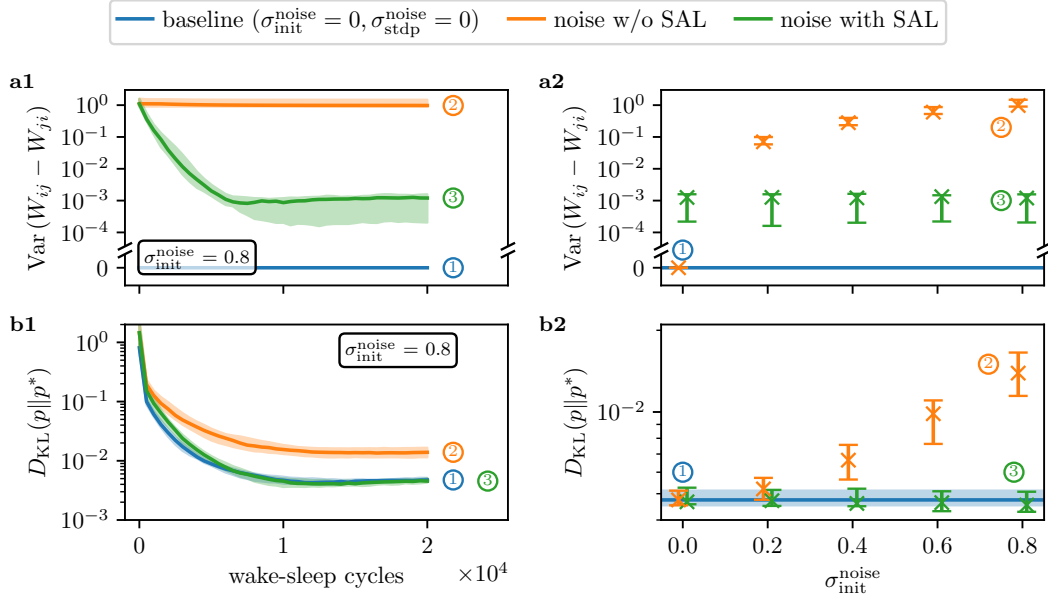
$$\Delta W_{ij} = \eta [\langle z_i z_j \rangle_{\text{wake}} - \langle z_i z_j \rangle_{\text{sleep}}], \quad (4)$$

where  $\eta$  denotes the learning rate and  $\langle \cdot \rangle_x$  the expectation value in the respective phase. Note that because of  $\langle z_i z_j \rangle = \langle z_j z_i \rangle$ , wake-sleep will always produce symmetric weight updates.

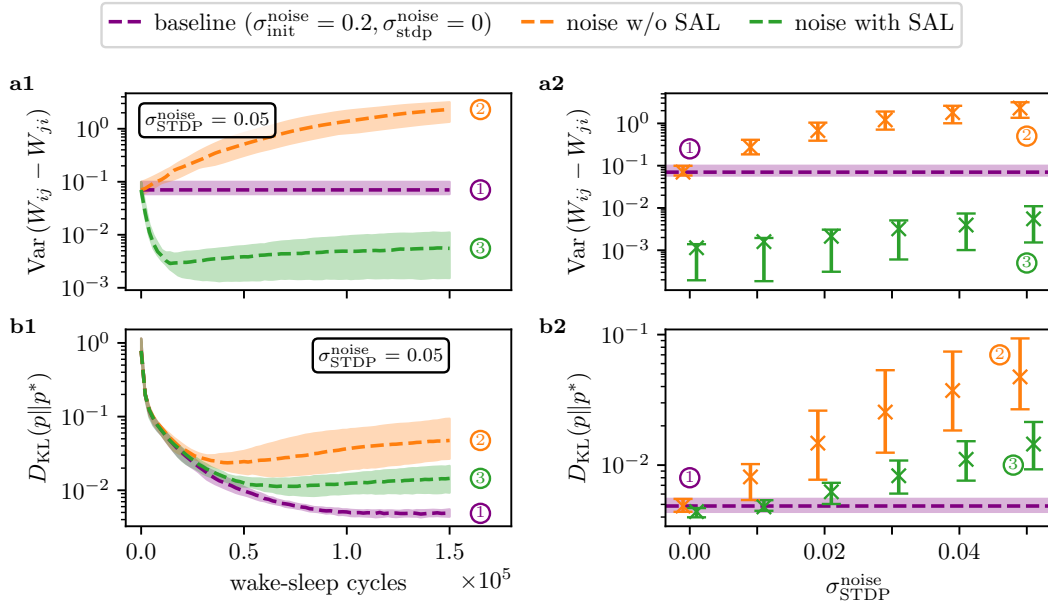


**Figure 3: Working principle of SSNs.** a) An SSN consists of a recurrent spiking network with symmetric reciprocal connections in theory, but asymmetric ones in practice. b) Each neuron fires stochastically as a function of the membrane potential (blue), which gives rise to a sampling process from an underlying distribution  $p(\mathbf{z})$ . The refractory state of each neuron is mapped to a binary variable  $z \in [0, 1]$  (green and red). STDP with a left-right symmetric window (orange) is used to implement spike-based wake-sleep. Because the synaptic update is local to each synapse, reciprocal weight updates are also asymmetric. c) Example sampled distribution for  $N = 4$  neurons.

In the framework of SSNs, the estimation of  $\langle z_i z_j \rangle_x$  can be carried out by STDP as introduced in eq. (2): We use left-right-symmetric, triangular STDP windows with  $f(\Delta t) = \max \left( -\frac{\Delta t}{\tau_{\text{ref}}} - 1, 0 \right)$  and  $\alpha_{a/c} = 1$  for the wake and  $\alpha_{a/c} = -1$  for the sleep phase to get an approximation of  $\langle z_i z_j \rangle_x$  purely by observing the spikes. A global gating signal can induce the phase switching [65]. To distinguish between eq. (4) and its bio-plausible version relying on STDP, we will refer to the former as *state-based* wake-sleep and to the latter as *spike-based* wake-sleep; throughout this work, we use spike-based wake-sleep learning. Because of synapse-specific parameter variations in physical substrates, spike-based wake-sleep can be subject to noise, which we model as synapse-specific variations of  $\alpha_{a/c}$ . The biases are trained



**Figure 4: Synaptic noise in SSNs.** An SSN is initialized with a symmetric weight matrix and trained with pure STDP-based wake-sleep (blue curve), which serves as a baseline (no noise). We also train a noised weight matrix (additive Gaussian noise with standard deviation  $\sigma_{\text{init}}^{\text{noise}}$ ) with (green) and without (orange) a SAL phase in addition to STDP-based wake-sleep learning. **a1**) Evolution of the variance of the weight differences  $W_{ij} - W_{ji}$  as a measure for the weight asymmetry for the three cases after each epoch. **a2**) Variance of the weight differences after training. **b1**) Evolution of the Kullback-Leibler divergence ( $D_{\text{KL}}$ ) during training between the freely sampled model distribution  $p$  and the target distribution  $p^*$ . **b2**) Final  $D_{\text{KL}}$  after training as a function of the initial noise variance.



**Figure 5: Plasticity noise in SSNs.** The basic configuration of fig. 4 is repeated. In addition to an initial weight noise of  $\sigma_{\text{init}}^{\text{noise}} = 0.2$ , we now also add noise  $\sigma_{\text{stdp}}^{\text{noise}}$  to the STDP kernel. For comparison, we show a baseline with  $\sigma_{\text{init}}^{\text{noise}} = 0.2$ , but without STDP kernel noise. **a1**) Evolution of the weight asymmetry  $\text{Var}(W_{ij} - W_{ji})$  for the three cases. **a2**) Weight asymmetry after training as a function of the STDP noise amplitude. **b1**) Evolution of the  $D_{\text{KL}}$  during training. **b2**)  $D_{\text{KL}}$  after training as a function of  $\sigma_{\text{stdp}}^{\text{noise}}$ . In all panels, we report the median and interquartile range over 20 different seeds.



with  $\Delta b_i = \eta[\langle z_i \rangle_{\text{wake}} - \langle z_i \rangle_{\text{sleep}}]$ , which boils down to a measurement of the firing rate.

When integrating the SSN model with the STDP version of wake-sleep on noisy physical substrates, we encounter the challenges discussed in section 1: First, the network must be initialized with symmetric weights, which requires the transport of effective weights between synapses. Any initial weight asymmetry will persist if symmetric weight updates are applied. Second, the STDP mechanism for the estimation of  $\langle z_i z_j \rangle$  is located at a specific synapse  $W_{ij}$  and is subject to parameter variations that are, in general, not identical to those of  $W_{ji}$ . Therefore, the STDP measurements for  $\langle z_i z_j \rangle$  and  $\langle z_j z_i \rangle$  will differ and thereby violate the requirement for symmetric weight updates during wake-sleep. Asymmetric weight updates will lead to diverging weight pairs, which distort the learned distribution and can even unlearn previously learned representations.

We address these issues by introducing an additional SAL update that co-occurs during the sleep phase. SAL quickly levels initial weight asymmetry and facilitates learning in noisy STDP scenarios by counterbalancing weight divergence. This stabilizes learning, particularly in lifelong online learning settings. It removes any assumption of implicit information exchange between synapses, ensuring that wake-sleep is truly local.

To illustrate the effect of SAL, we conduct two types of experiments, in which we consecutively introduce different types of noise: the *synaptic noise scenario* and the *plasticity noise scenario*. In both scenarios, we train a network of  $N = 7$  neurons to approximate a target Boltzmann distribution  $p^*$ .

**Synaptic noise scenario:** The first experiment (fig. 4) simulates fixed pattern noise on the synaptic weights by randomizing the initial weight matrix. To do so, we add Gaussian noise with variance  $\sigma_{\text{init}}^{\text{noise}}$  to the weight matrix. All synapses share identical STDP parameters ( $\alpha_{a/c} = \pm 1$ ), making the learning process equivalent to state-based wake-sleep.

SAL demonstrates a rapid capability to symmetrize weights, and is able to recover optimal training performance (fig. 4a1). This is consistent across various noise levels and weight differences (fig. 4a2). The ability of an SSN to learn target distributions is found to be critically dependent on the symmetry of reciprocal weights. Even minor deviations from symmetry hinder learning (fig. 4b1,b2). By applying SAL in alternating phases with functional plasticity (wake-sleep), we demonstrate that it does not impede learning speed (fig. 4b1). Consequently, SAL enables recovery of training performance comparable to the noise-free baseline (blue) (fig. 4b2).

**Plasticity noise scenario:** The second experiment (fig. 5) introduces noise to both the initial weight matrix and the wake-sleep STDP kernels. I.e., an initial noise value of  $\sigma_{\text{init}}^{\text{noise}} = 0.2$  is used; additionally, a random variable  $\xi$  is drawn from a truncated Gaussian distribution for each synapse to model STDP inhomogeneity ( $\alpha_{a/c} = \pm 1 \pm \xi$ ).

Even small discrepancies in STDP parameters for wake-sleep lead to weight divergence in reciprocal synapses (fig. 5a1). Without regularization, this divergence can cause unlearning of distributions (fig. 5b1). SAL effectively counteracts the tendency of reciprocal weights to di-

verge, maintaining alignment throughout the training process (fig. 5a1,a2). Consequently, wake-sleep learning is significantly improved with SAL, facilitating a more accurate learning of target distributions even when symmetric wake-sleep plasticity remains active (fig. 5a1,a2). However, due to the presence of the wake-sleep drift, the baseline performance cannot be fully recovered.

## 2.4.2 Application 2: Cortical microcircuits

We now consider a specific implementation of SAL in dendritic cortical microcircuits. Such units have been suggested to enable the transmission of forward signals and backward errors across cortical hierarchies, realizing a biologically plausible variant of gradient descent through BP [29]. To our knowledge, our implementation represents the first spiking version of this microcircuit model, thereby offering an alternative implementation of spike-based backpropagation to burstprop [32].

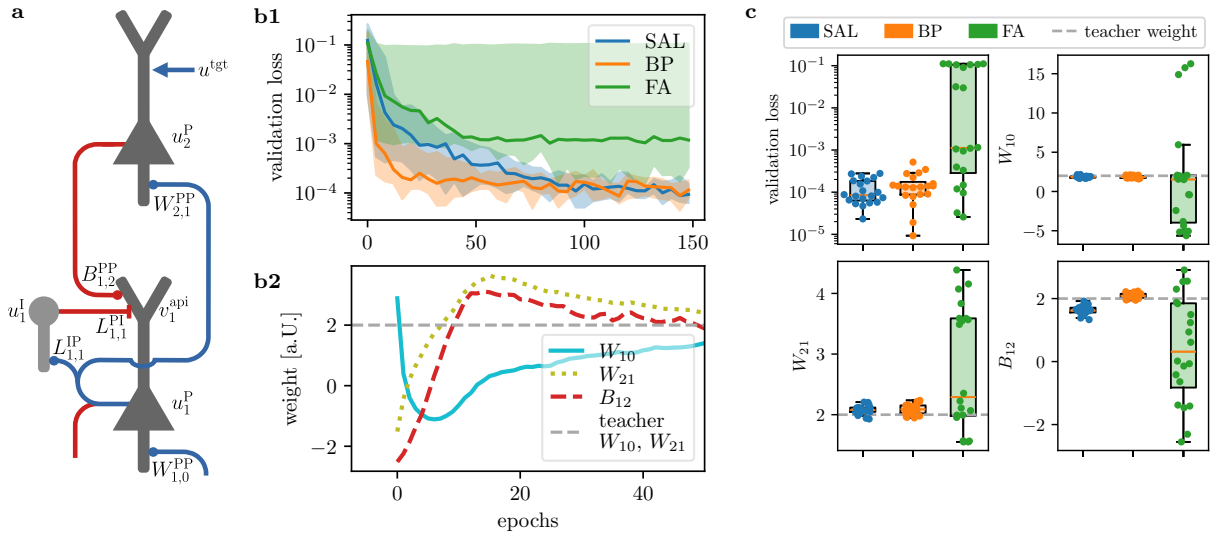
Each microcircuit comprises two types of neuron populations: pyramidal cells and interneurons. These neurons are organized into layers that correspond to cortical areas, following a biologically plausible connectivity pattern (see fig. 6a).

**Student-teacher task** In the following, we explore the model in a student-teacher task. Here, a chain of two pyramidal neurons is trained with SAL and, for comparison, with FA and BP as a reference. A teacher configuration of the same size and depth is set up with fixed target weights to produce a non-linear input-output mapping. The objective for the student circuits is to learn this function by adapting their weights accordingly.

A crucial aspect of this task is that successful learning requires the simultaneous adaptation of all the synapses in the network. For the latent layer, it is essential that a meaningful error signal is transmitted through the feedback connections. To illustrate the limitations of FA, we conducted experiments where the feedback weights were randomly initialized; i.e., in  $\sim 50\%$  of experiments, feedback weights were initialized with opposite sign compared to feed-forward weights. Consequently, in these scenarios, the transported error also had the wrong sign, causing weight updates in the incorrect direction, and resulting in high average losses with a very large spread towards even higher values (fig. 6b1) and a drive of the weights away from their target (fig. 6c, green).

In contrast to the previous experiments with SSNs, here SAL was only applied to the backward weights  $\mathbf{B}_{1,2}^{\text{PP}}$ . This allows  $\mathbf{B}_{1,2}^{\text{PP}}$  to quickly align with  $\mathbf{W}_{2,1}^{\text{PP}}$  at the start of training and subsequently follow its target, as shown in fig. 6b1,b2. This facilitates fully local learning from any initial weight configuration: in fig. 6a2,  $\mathbf{B}_{1,2}^{\text{PP}}$  is initially set with the incorrect sign (indicated by the red dashed line), which should have been positive. As a result,  $\mathbf{W}_{1,0}^{\text{PP}}$  receives an erroneous error signal and decreases its value during the first 5 epochs. With SAL,  $\mathbf{B}_{1,2}^{\text{PP}}$  learns to eventually switch to the correct sign. From this moment on, a meaningful error signal is induced in  $\mathbf{v}_1^{\text{api}}$ , enabling  $\mathbf{W}_{1,0}^{\text{PP}}$  to converge to its target value (indicated by the gray dashed line).

The study highlights the importance of correct error signal transmission through feedback connections for successful learning, demonstrating the limitations of FA and the effectiveness of SAL in aligning backward connections with



**Figure 6: SAL enables accurate BP in a spiking cortical microcircuit model.** **a)** Cortical microcircuit model for biologically plausible error backpropagation based on [29], which we augment with a spiking mechanism. **b1)** Performance of different learning rules on a teacher mimic task. SAL outperforms FA, and training performance is on par with weight copying (BP). Note the large performance variability of FA, which depends critically on the initial weights. **b2)** Evolution of top-down and bottom-up weights during SAL learning for a case where initial feedback weights carry the wrong sign. Forward weights  $W_{1,0}^{PP}$  first evolve in the wrong direction (with FA, they would explode, see h), but the gradual alignment of the backward weights  $B_{1,2}^{PP}$  improves the feedback error signals, ultimately recovering the performance of vanilla BP. **c)** Distribution of final validation loss and learned weights in the student network.

feedforward weights, thus ultimately facilitating fully local BP learning.

## 2.5 Other PSP shapes and their effect

So far, we have only considered rectangular PSP shapes. More biologically plausible PSPs exhibit a more complex behavior when using SAL, effectively introducing a bias-dependent behavior in the symmetrization (fig. 7). The bias serves as a proxy for the base firing rate, which can result from neuron-specific leak potentials or different input rates from the surrounding networks, averaged over time. In the following, we discuss how SAL behaves in complex networks when other PSP shapes are used.

We turn back to the two neuron system of section 2.2. As the most important feature for evaluating the effectiveness of SAL, we assess the shape of the basin of attraction in the  $W_{ij}/W_{ji}$  plane as visualized by the phase plane diagram in fig. 2e. The basin of attraction provides information about the weight pairs to which SAL converges.

We employ an  $\alpha$ -shaped kernel

$$\kappa(t) = \Theta(t) \frac{\tau_{\text{ref}}}{\tau_{\text{syn}}^2} t \exp\left(-\frac{t}{\tau_{\text{syn}}}\right), \quad (5)$$

with different synaptic constants  $\tau_{\text{syn}}$  (see fig. 7a) to model PSPs often found in biological neurons and used in neuromorphic hardware. These are compared with the rectangular kernel used in previous results.

In fig. 7c, we visualize the attractors of the rectangular and  $\alpha$ -shaped PSPs with a short and long time constant  $\tau_{\text{syn}}$ . For the sake of clarity, we do not plot the flow field, but concentrate on the shape of the attractors. Notably, SAL converges in all cases, i.e., all fixed points are indeed stable and remain organized as line attractors, ensuring that SAL gives rise to useful weight updates. For a formal proof, we refer

to section 4.2.2. Furthermore, the signs of the weights are matched in all cases (i.e.,  $\text{sgn}(W_{ij}) = \text{sgn}(W_{ji})$ ), independently of the biases and the PSP shape<sup>2</sup>. This is an important result, as sign-conserving FA has been shown to permit learning performance close to BP even without matching weight amplitudes [40].

These deviations are best understood by considering the effect of PSP shapes on the STDD (fig. 7b). Here, we provide a qualitative explanation, but refer to section 4.2.1 for the exact analytical result.

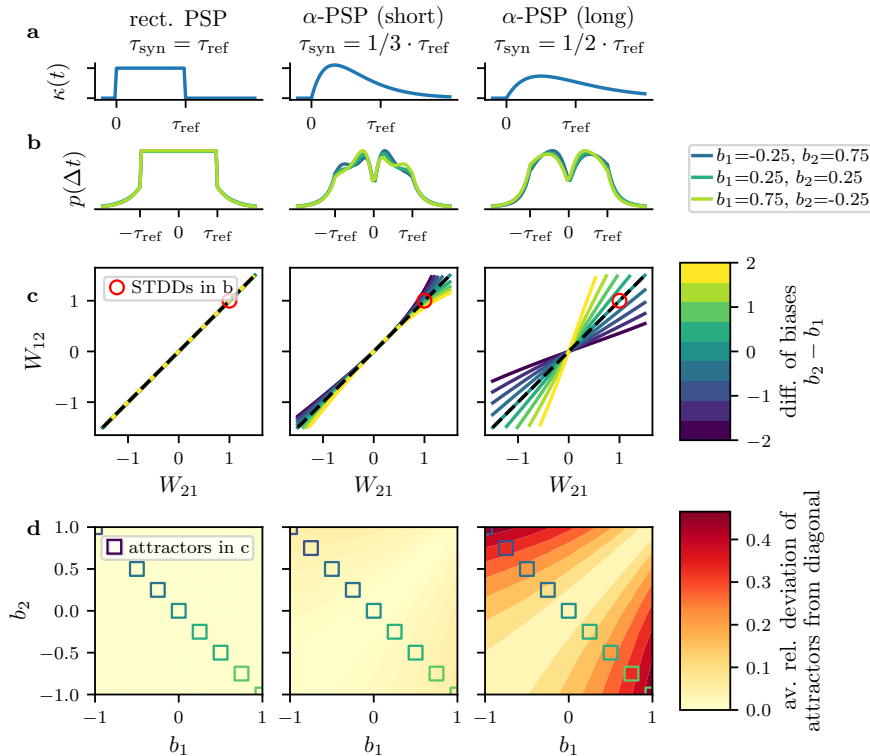
Since the firing probability is a monotonic function of the membrane potential, we expect the shape of the STDD to roughly follow that of the PSPs. Indeed, for rectangular PSPs, we observe a flat distribution, as a constant PSP leads to a uniform spiking probability in the postsynaptic neuron  $j$  for all  $\Delta t < \tau_{\text{ref}}$ . On the other hand, for  $|\Delta t| > \tau_{\text{ref}}$ , the spike timing differences are produced by two independent Poisson processes, giving rise to a left-right symmetric exponential distribution.

For  $\alpha$ -PSPs, the STDD shape is still similar to that of the PSP, but the strict left-right symmetry is broken. We identify two main causes for this effect.

First, we note that  $\alpha$ -PSPs extend beyond the refractory period and can therefore stack, unlike the rectangular PSPs with  $\tau_{\text{syn}} = \tau_{\text{ref}}$ . The probability of such stacking increases with the presynaptic partner's firing rate, which, in turn, is determined by its bias. Unequal biases lead to asymmetric stacking probabilities and therefore to asymmetric STDDs. Furthermore, this stacking also means that the spiking processes are not independent anymore for  $|\Delta t| > \tau_{\text{ref}}$ . However, this phenomenon has a much smaller effect due to the exponential decay of the SAL STDP kernel. Evidently, these stacking effects are altogether more pronounced for longer

<sup>2</sup>We will discuss what sign changes mean in networks subject to Dale's law in section 3.





**Figure 7: SAL works with different PSP shapes.** **a)** We consider a two-neuron system together with three common PSP kernels. **b)** STDDs at  $W_{12} = W_{21} = 1$  for three different bias combinations. The STDDs of the  $\alpha$ -kernels show a slight asymmetry if  $b_1 \neq b_2$ . **c)** Phase plane diagram of reciprocal weights for nine bias combinations. For clarity, only the attractors are shown. The biases used here are marked as squares in the respective color in panel d). Rectangular PSPs always yield perfect symmetrization (all attractors are located on the diagonal). On the other hand, symmetrization is not perfect for  $\alpha$ -PSPs. The attractor deviates from the diagonal if the difference in biases is large, with longer PSPs causing larger deviations. Importantly, however, the sign of the weights is always aligned correctly. Red circles indicate the weights used for the STDDs above. **d)** Average relative deviation of the attractor from the diagonal as a function of biases  $b_1$  and  $b_2$ . Rectangular PSPs symmetrize perfectly for all biases. Short PSPs (middle) only cause small deviations (less than 10% across the whole bias-space), while long PSP tails incur larger deviations (right).

PSPs, and thus represent the main reason for the discrepancies in fig. 7b-d, right panels.

The second cause of STDD symmetry breaking is best observed in fig. 7b, middle panel, where a second bump around  $\pm\tau_{\text{ref}}$  stands out. This is an “echo” of the PSP’s peak: If neuron  $i$  has a high bias, the probability for it to spike again (independently of neuron  $j$ ) immediately after the end of its refractory period, i.e., after approximately  $\tau_{\text{ref}}$  is also raised. If now a spike in neuron  $j$  is elicited by the first PSP from neuron  $i$ , a causal  $\Delta t$  is followed by its anticausal “echo” at  $\Delta t' \approx \Delta t - \tau_{\text{ref}}$ . Since this effect is also bias-dependent, it also breaks the STDD symmetry for unequal neuronal biases. Being most pronounced for short, peaked PSPs, it represents the main cause behind the deviations in fig. 7b-d, middle panels.

Finally, we quantify the deviation of the attractor from the diagonal by computing the average relative distance to the diagonal as a function of  $b_1$  and  $b_2$  (see also section 4.5). In general, the relative deviation is smaller for shorter PSPs, as well as for smaller biases (fig. 7d, middle and right panel).

This indicates an effective method for further improving synaptic alignment if vanilla SAL is not sufficient. By introducing a further alignment phase into the learning procedure during which functional plasticity is absent and neuronal biases are sufficiently reduced (for example, through negative input currents or background inhibition), SAL can always be

ensured to perform weight symmetrization to the required precision even when neuronal biases remain diverse.

### 3 Discussion

Plausible algorithms for physical computing – whether instantiated in neuromorphic hardware or biological brains – need to tackle the weight transport problem. In this paper, we have presented SAL, a fully local algorithm for effective weight transport in spiking neuronal networks based on STDP.

SAL enables the alignment of bottom-up and top-down information flow (for instance, of the sensory information and error signals, depending on the computational paradigm [66–68]), thereby fulfilling the core requirement of credit assignment in physical systems: the locality of information in space and time. In the domain of neuromorphic computing, SAL can play a key role in the implementation of fully local on-chip learning schemes. In models for computation in the brain, SAL can increase the biological plausibility when networks are subject to symmetry constraints. Additionally, we emphasize SAL’s ability to increase the robustness of networks to the omnipresent fixed pattern noise found in any physical substrate. More concretely, SAL automatically balances effective weights and does not just copy the

numerical value. Thus, the effect of morphological variance of different neurons on computation is also compensated.

In contrast to other approaches, SAL is free of any (oftentimes implicit) assumption regarding the symmetry of shared parameters between the involved neurons and synapses: Instead, SAL works independently in each synapse and compensates for asymmetric weight updates. Moreover, SAL uses the omnipresent temporal noise in physical substrates as a resource: While noise in neural network is often regarded as destabilizing for computation, it actually drives the stabilizing effect of SAL. Finally, we have also shown that SAL is mostly agnostic to the underlying network topology. It is compatible with a plethora of network architecture that involve reciprocal connectivity. In this context, we also highlight that it has allowed us to demonstrate the first functional spiking implementation of the backpropagating microcircuits proposed in [29].

**Related work** Broadly, existing approaches to address the weight transport problem can be divided into four categories:

First, some studies simply assume weight symmetry in their theoretical derivations and implement physically implausible copy operations in their simulations. While this allows to show that the respective algorithms are in general capable of successfully performing credit assignment, the absence of a solution to the weight transport problem calls their biological plausibility into question, while also limiting their applicability for neuromorphic on-chip learning. An example here is equilibrium propagation [28] or the work by Xie and Seung [27].

Second, random, fixed backprojections as in FA [38, 69] are often cited as a one-fits-all approach to solve the weight transport problem. However, numerous studies have highlighted its limitations when it comes to scaling to deeper networks or robustness [40, 41]. The detrimental effect of misaligned weights carries over to spiking networks, as we have shown here. Importantly, it is known that weight alignment does not need to be perfect for transportation of meaningful gradients in deep networks [40, 41, 70]. Instead, the conservation of the sign is usually sufficient, which SAL guarantees for the most widely assumed PSP shapes.

The third category is formed by algorithms that are based on weight decay. The idea was first introduced by Kolen and Pollack [31], using a constant weight decay that causes the network to gradually forget its asymmetric initial state. However, symmetrization then relies on strictly symmetric weight updates, and it is not clear how these symmetric updates can come about given heterogeneous synapses and plasticity; in that sense, weight decay algorithms only relegate the weight transport problem to a weight *update* transport problem. By contrast, SAL relaxes all implausible constraints on symmetric updates because it is able to dynamically realign weights during training.

In the fourth category, we summarize algorithms that dynamically learn useful reciprocal weights in a physically plausible manner. With (difference) target propagation, [71–74] learn pseudo-inverse backprojections to perform Gauss-Newton optimization. For error backpropagation, [25, 26, 41, 75] learn feedback weights that approximately align with the forward Jacobian, but are data-specific and do not necessarily match the forward weights in angle or amplitude.

A particular example from this class of algorithms is PAL [41], as it can be regarded as the rate-based complement to SAL. Notably, their common denominator is the functional role of noise: In PAL, the information carrying signal is augmented by high frequency noise, which moves in loops through the network. On its way, the noisy signal is affected by the synaptic weights and therefore carries information that can be exploited locally to align weights. A further commonality between SAL and PAL is their ability to train all backprojections in a network at the same time and without disturbing forward weight learning. Note, however, that all the aforementioned approaches are quintessentially rate-based, i.e., they are based on time-continuous information exchange between neurons. SAL, on the other hand, is designed from the ground up for spiking networks, where discrete, sparse events are the central information carrier.

In [42], a spiking model for autoencoders using STDP was introduced. Similarly to SAL, an anti-Hebbian window (called “mirrored STDP”) is employed to learn the backward projections of the autoencoder. Despite the apparent similarity between SAL and mirrored STDP, the two plasticity rules actually serve a different purpose: In [42], the combination of Hebbian and anti-Hebbian learning produces the same weight updates in the forward and backward direction. In this sense, it is a spiking, biologically plausible implementation of the Kolen-Pollack algorithm [31]. As a result, mirrored STDP cannot equilibrate initial weight differences. Furthermore, it is not discussed whether the model can compensate for parameter noise on neuronal and plasticity parameters, which is a crucial feature of SAL and decisive for its efficacy in neuronal systems with analog components, whether biological or neuromorphic.

A different approach to spike-based credit assignment is proposed in [76]. Following the energy-based paradigm of equilibrium propagation, O’Connor et al. propose a spiking stochastic approximation thereof. While the authors demonstrate the feasibility of their approach, they also acknowledge that it lacks a solution to the weight transport problem; this is exactly the gap which SAL is able to fill, and a combination of SAL with Spiking Equilibrium Propagation could shed light on neuronal credit assignment and drive efficient hardware implementations.

BurstProp [32], a solution to credit assignment in layered spiking networks, uses a local plasticity rule together with postsynaptic bursts to determine the sign of the synaptic change of the forward weights. However, the authors implement FA for the transportation of the top-down teaching signal; as above, BurstProp may be combined with SAL to improve biological plausibility and learning performance.

**Applications and constraints** Aside from computation in the brain, SAL specifically targets spiking neuromorphic systems, especially if they contain analog components. Such systems are prone to substrate variability and parameter drift, which destabilize training performance and limit real-world applicability. This applies, for example, to emulations of quantum systems on analog spiking hardware [21, 22] or Bayesian inference using spiking nanolasers [77]. In all of these cases, asymmetries due to substrate variability constitute the main limiting factor for the task performance of the respective networks.

Importantly however, STDP-like learning forms the standard for on-board/on-chip learning on many platforms [20,

58, 78–82]. Therefore, SAL may provide the missing link for stable, physical learning systems: by facilitating a fully local implementation of effective weight sharing, it can compensate parameter noise and drift. Additionally, it paves the way for a fully local, analog implementation of BP.

This capability positions SAL as a robust solution for adaptive systems, such as smart sensors and wearable devices, which must cope with challenges like sensor degradation or environmental changes while maintaining efficiency and autonomy. In this context, we expect that memristive devices [58, 82], known for their high density and low power consumption, provide an ideal substrate for implementing SAL’s learning rules.

Complementary to this, SAL is designed to address weight transport in spiking theories of the brain. It suggests a functional need for diverse STDP curves, thus providing an explanation for their observation in nature [12, 83].

However, contrary to many hardware platforms [20, 79, 80], biological synapses cannot simply switch their sign, as they typically express only one type of neurotransmitter. This makes biological synapses either excitatory or inhibitory, a phenomenon known as Dale’s law [84]. So far, we have neglected this aspect in all discussions and simulations of our method. However, an important property of SAL is its ability to change the sign of a given synapse. To align SAL with Dale’s principle, we point out that (excitatory) pyramidal neurons in the cortex are not only connected directly (via purely excitatory connections), but also via inhibitory interneurons. The resulting *effective* weight is thus the sum of the inhibitory and direct excitatory weights. By modulating only the weight of the direct excitatory synapse, the whole range from effective inhibition to excitation of the postsynaptic neuron can be covered. In such circuits, SAL can act only on the direct excitatory pathway, but still align the total effective weights between the forward and backward paths.

In summary, we contend that symmetry is not just a mere token of theoretical elegance, but also an important ingredient for practical deployment across a wide range of models and applications. Instead of working around it when faced with its absence, SAL offers a solution for recovering and maintaining symmetry in real physical systems, by making use of noise as a fundamental computational ingredient.

## 4 Methods

As in the main text, bold lowercase variables  $\mathbf{x}$  denote vectors, bold uppercase letters  $\mathbf{X}$  matrices. We denote spike times of neuron  $i$  as  $t_i$ . The most recent spike emitted by neuron  $j$  before the time  $t$  is denoted as  $t'_j$ . The set of all past spikes from neuron  $i$  is denoted by  $\{t_i\}$ . The output spike train from neuron  $i$  is

$$S_i(t) = \sum_{\{t_i\}} \delta(t - t_i), \quad (6)$$

where  $\delta(x)$  is the Dirac delta distribution.

### 4.1 Neuron model

We use a generalized linear model (GLM) [85] to model the dynamics of spiking neurons. The membrane potential of

neuron  $k$  is given by

$$u_i(t) = b_i + \sum_k W_{ik}(t) \int_0^\infty \kappa(s) S_k(t-s) ds, \quad (7)$$

where  $b_i$  is the neuron’s bias, which can be associated with the leak potential,  $\kappa(t)$  the kernel of the post-synaptic potential (PSP),  $S_k(t)$  the spike trains coming from the presynaptic neuron  $k$ , and  $W_{ik}$  the corresponding synaptic weight. If not stated otherwise, we employ a rectangular PSP kernel,

$$\kappa(t) = \Theta(t) - \Theta(t - \tau_{\text{syn}}), \quad (8)$$

where  $\tau_{\text{syn}}$  denotes the synaptic time constant.

The output spikes of a neuron are generated by an inhomogeneous Poisson process with absolute refractory period of length  $\tau_{\text{ref}}$ . The spiking probability in the time interval  $[t; t + dt]$  is given by

$$p(t_i|t'_i) = \begin{cases} r_i(t) dt & \text{if } t - t'_i > \tau_{\text{ref}} \\ 0 & \text{else,} \end{cases} \quad (9)$$

where

$$r_i(t) = \tau_{\text{ref}}^{-1} \exp(u_i(t)) \quad (10)$$

is the instantaneous firing rate, rescaled by the refractory period. The timescale of the rectangular PSP kernel is matched with the length of the refractory period  $\tau_{\text{syn}} = \tau_{\text{ref}}$ .

We choose this model to account for stochastic firing pattern found throughout networks in the brain and to model the presence of temporal noise in the nervous system while being mathematically tractable. It can be shown that leaky integrate and fire (LIF) neurons in the high-conductance driven by high-frequency spike noise state can have similar statistical properties as the model presented above [62].

### 4.2 Spike-based alignment learning

The working principle of spike-based alignment learning (SAL) builds on the observation that weights leave a characteristic imprint on the distribution of spike timing differences of pre- and postsynaptic spikes. In section 2.3 and figs. 2 and 7, we have given a qualitative understanding of our weight alignment mechanism, which we now follow up with a more detailed explanation.

Typically, unequal reciprocal weights  $W_{ij} \neq W_{ji}$  produce skewed/asymmetric spike-timing difference distributions (STDDs). This is informative of the weight difference and can be exploited for symmetrization. This asymmetry can be extracted by a standard spike-timing-dependent plasticity (STDP) rule, which updates  $W_{ij}$  after every occurrence of a nearest-neighbor spike pair with  $\Delta t_{ij} = t'_i - t'_j$  via

$$\Delta W_{ij} = \eta \left( \underbrace{\Theta(-\Delta t_{ij}) \alpha_a f^-(\Delta t_{ij})}_{\text{anti-causal}} + \underbrace{\Theta(\Delta t_{ij}) \alpha_c f^+(\Delta t_{ij})}_{\text{causal}} \right). \quad (11)$$

Here,  $f(\Delta t)$  defines the shape of the learning window, which we choose to be  $f(\Delta t) = \exp(t/\tau)$  in accordance with [86].  $\alpha_c$  ( $\alpha_a$ ) is the prefactor for the (anti)causal branch, which we set to be +1 (−1) to strengthen (weaken)  $W_{ij}$  if a (anti)causal spike pair occurs.

In practice, SAL works with spike pairs of infinite-range as long as the number of causal and anti-causal  $\Delta t$  are the same; i.e. on average, a spike contributes as many times

to causal  $\Delta t$  as to anti-causal ones; if this is not the case, weights do not necessarily converge to a fixed point when using SAL. In fact, the experiments in section 2.4 use correlations of *all* spike pairs, demonstrating that SAL also works beyond the strict analytical regime of nearest neighbor spike pairs. The infinite-range STDP rule can be written as

$$\begin{aligned} \dot{W}_{ij}(t) = & \eta (S_j(t) \int_0^\infty \alpha_a(s) f^-(s) S_i(t-s) ds \\ & + S_i(t) \int_0^\infty \alpha_c(s) f^+(s) S_j(t-s) ds) \end{aligned} \quad (12)$$

#### 4.2.1 Analytical calculation of the spike-timing difference distribution

Our analytical computation of the STDD builds on the insight that a time discretized version of the GLM can be expressed as a discrete-time Markov chain. The following derivation is inspired by [61]. We introduce a discrete counting variable  $\zeta_i \in \mathbb{N}$ , that defines the state of neuron  $i$  and a state vector  $\zeta \in \mathbb{N}^N$  collecting the states of all  $N$  neurons in the network. Due to the Markov property, the network state  $\zeta'$  of the next time step depends only on the current state  $\zeta$ .

$\zeta_i$  counts the number of time steps since neuron  $i$  has spiked the last time. Depending on its own state  $\zeta_i$  and the states of the other neurons  $\zeta_k$ , neuron  $i$  can spike with some probability, for which we set  $\zeta'_i = 1$ . If it doesn't spike, the counter is incremented,  $\zeta'_i = \zeta_i + 1$ .

For each neuron, we define analogous to eq. (7) a membrane potential

$$u_i = b_i + \sum_k W_{ik} \kappa(\zeta_k), \quad (13)$$

where  $b_i$  is the neuron's bias,  $W_{ik}$  the synaptic weight and  $\kappa(\zeta_k)$  the PSP induced by the last spike from neuron  $k$ .

The update rules of the Markov chain is determined by the transition operator  $T(\zeta'|\zeta)$  which describes the probability to obtain a certain state  $\zeta'$  given the current state  $\zeta$ . The transition operator  $T(\zeta'_i|\zeta)$  for a single neuron is given by:

$$T(\zeta'_i = 1|\zeta) = \begin{cases} r_i & \text{for } \zeta_i > \tau_{\text{ref}} \quad (\text{spike}) \\ 0 & \text{else} \end{cases} \quad (14a)$$

$$T(\zeta'_i = \zeta_i + 1|\zeta) = \begin{cases} 1 - r_i & \text{for } \zeta_i > \tau_{\text{ref}} \quad (\text{no spike}) \\ 1 & \text{else} \end{cases} \quad (14b)$$

All other transitions are forbidden. As before,  $\tau_{\text{ref}}$  represents the refractory period, but here as a natural number in terms of time steps. We use a spiking probability  $r_i = (1 + \exp(-(u_i - \log \tau_{\text{ref}})))^{-1}$ . Because  $u_i$  does not depend on  $\zeta_i$  itself, all neurons can be updated in parallel.

Starting from the transition rules eq. (14) for one neuron, we can construct the transition rules for a two neuron system  $\zeta^{(2)} = (\zeta_1, \zeta_2)$ . In this case, if neuron 1 is not refractory, it can spike with a probability of  $r_1$  derived from  $u_1 = b_1 + W_{12}\kappa(\zeta_2)$ . Respectively, neuron 2 spikes with a probability determined by  $u_2 = b_2 + W_{21}\kappa(\zeta_1)$ . The resulting nine possible transitions are:

- neuron 1 and 2 are refractory ( $\zeta_1 < \tau_{\text{ref}}$  and  $\zeta_2 < \tau_{\text{ref}}$ ):

$$T(\zeta'_1 = \zeta_1 + 1, \zeta'_2 = \zeta_2 + 1|\zeta_1, \zeta_2) = 1 \quad (15a)$$

- neuron 2 is refractory ( $\zeta_2 < \tau_{\text{ref}}$ ); 1 can spike ( $\zeta_1 \geq \tau_{\text{ref}}$ ):

$$T(\zeta'_1 = 1, \zeta'_2 = \zeta_2 + 1, |\zeta_1, \zeta_2) = r_1 \quad (15b)$$

$$T(\zeta'_1 = \zeta_1 + 1, \zeta'_2 = \zeta_2 + 1, |\zeta_1, \zeta_2) = (1 - r_1) \quad (15c)$$

- neuron 1 is refractory ( $\zeta_1 < \tau_{\text{ref}}$ ); 2 can spike ( $\zeta_2 \geq \tau_{\text{ref}}$ ):

$$T(\zeta'_1 = \zeta_1 + 1, \zeta'_2 = 1, |\zeta_1, \zeta_2) = r_2 \quad (15d)$$

$$T(\zeta'_1 = \zeta_1 + 1, \zeta'_2 = \zeta_2 + 1, |\zeta_1, \zeta_2) = (1 - r_2) \quad (15e)$$

- neuron 1 and 2 can spike ( $\zeta_1 \geq \tau_{\text{ref}}$  and  $\zeta_2 \geq \tau_{\text{ref}}$ ):

$$T(\zeta'_1 = 1, \zeta'_2 = 1|\zeta_1, \zeta_2) = r_1 r_2 \quad (15f)$$

$$T(\zeta'_1 = 1, \zeta'_2 = \zeta_2 + 1, |\zeta_1, \zeta_2) = r_1 (1 - r_2) \quad (15g)$$

$$T(\zeta'_1 = \zeta_1 + 1, \zeta'_2 = 1, |\zeta_1, \zeta_2) = (1 - r_1) r_2 \quad (15h)$$

$$T(\zeta'_1 = \zeta_1 + 1, \zeta'_2 = \zeta_2 + 1, |\zeta_1, \zeta_2) = (1 - r_1) (1 - r_2) \quad (15i)$$

All other transitions are forbidden ( $T(\zeta'_1, \zeta'_2|\zeta_1, \zeta_2) = 0$ ).

With these transition probabilities (eq. (15)) we can compute STDD of the two neuron system. In the following derivation, we introduce a maximum value  $Z \gg \max(\tau_{\text{ref}}, \tau_{\text{syn}})$  for  $\zeta_i$  to limit the possible state space of our two neuron system to a finite number.

Let us define the probability state vector  $\pi \in \mathbb{R}^{Z^2}$ , that contains the probabilities for all states,

$$\pi = \left( p(\zeta_{1,1}^{(2)}), \dots, p(\zeta_{1,Z}^{(2)}), p(\zeta_{2,Z}^{(2)}), \dots, p(\zeta_{Z,Z}^{(2)}) \right)^T, \quad (16)$$

where we use the shorthand  $\zeta_{i,j}^{(2)} = (\zeta_1 = i, \zeta_2 = j)$ . Together with the transition matrix  $\mathbf{T} \in \mathbb{R}^{Z^2 \times Z^2}$  the evolution of the two neuron system from state  $\pi$  to  $\pi'$  can be computed via

$$\pi' = \mathbf{T}\pi. \quad (17)$$

$\mathbf{T}$  contains the transition probabilities defined in eq. (15) where the columns and rows are arranged in the same order as in  $\pi$ ,

For calculating the STDD, we are interested in the *invariant* or *steady state distribution* of the system, i.e. the distribution  $\pi^*$  which does not change when  $\mathbf{T}$  is applied. We obtain  $\pi^*$  by solving the eigenvalue problem

$$\lambda \pi^* = \mathbf{T}\pi^* \quad (18)$$

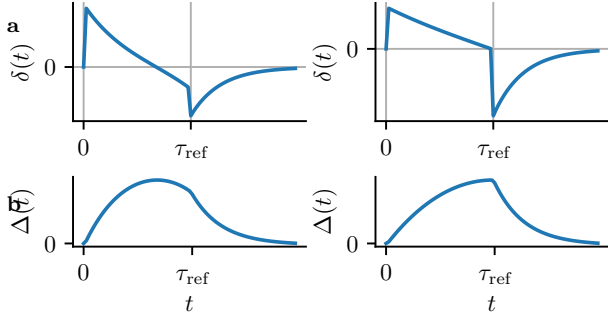
for the real eigenvalue  $\lambda = 1$ .

In the following, we are calculating the probability that neuron 2 spikes  $\Delta t$  time steps after neuron 1 (i.e. the *right* side of the STDD). To calculate negative  $\Delta t$  values (i.e. the *left* side), we can use the symmetry of the two neuron system and simply swap neuron 1 and 2 in the equations.

We start with a steady state vector  $\pi_{\text{right}}^*$ , where all entries for which  $\zeta_1 > 1$  are set to zero, i.e. we consider only those states and their probabilities, for which neuron 1 has just spiked.

Obtaining a  $\Delta t$  means that both neurons undergo  $\Delta t - 1$  transitions where they increment their  $\zeta$ s (as described by eqs. (15a), (15c), (15e) and (15i)). They are put together in the transition matrix  $\mathbf{T}_{\text{nospike}}$ , which is basically a version of  $\mathbf{T}$  in which all rows in which  $\zeta'_1 = 1$  or  $\zeta'_2 = 1$ .

After the  $\Delta t - 1$  steps, neuron 2 spikes for which we construct an additional matrix  $\mathbf{T}_{\text{spike}} \in \mathbb{R}^{Z \times Z^2}$  that contains



**Figure 8:** **a)** STDD deviation  $\delta(t)$  for  $W_{12} = W_{21} + \varepsilon$  (with  $\varepsilon = 0.005$ ) and rectangular PSPs, for positive (left) and negative weights (right). **b)** Integrated deviation  $\Delta(t)$  for positive (left) and negative (right) weights.

the transition probabilities eqs. (15d) and (15h). Hence, the right side of the STDD  $\mathbf{p}(\Delta t > 0)$  is given by

$$\mathbf{p}(\Delta t > 0) = \sum_{i=1}^{Z-1} \mathbf{T}_{\text{spike}} \mathbf{T}_{\text{nospike}}^{i-1} \boldsymbol{\pi}_{\text{right}}^*. \quad (19)$$

The whole distribution is obtained by concatenating the right and left distribution.

#### 4.2.2 Proof for the existence and stability of fixed points in SAL

To show the existence and stability of fixed points in SAL, we introduce the following notation: We use a  $\hat{\cdot}$  to indicate the time-mirrored version of a variable, i.e.,  $\hat{g}(t) := g(-t)$ . Furthermore, we use a  $\cdot^+$  to indicate the right part of a function, i.e. positive domain  $t \geq 0$ , and  $\cdot^-$  for the left part.

For the STDP kernel we assume  $f^+(t) > 0$ ,  $f^-(t) < 0$  and  $f^+(t) = -\hat{f}^-(t)$  as well as positive derivatives  $f^{+'}(t) = f^{-'}(t) > 0$ . The exact shape does not matter in our consideration.

We denote the STDD for a given set of symmetric weights  $W_{12} = W_{21}$  with  $p_0(t)$ . Additionally, we introduce a distribution  $p_\varepsilon(t)$  of a set of weights  $W_{21} = W_{12} + \varepsilon$  with a small  $\varepsilon \rightarrow 0$ . From this, we define for  $t \geq 0$  the deviation

$$\delta := p_\varepsilon^+ - p_0^+ - (\hat{p}_\varepsilon^- - \hat{p}_0^-). \quad (20)$$

Importantly, the STDD is defined such that it contains only nearest-neighbor spike pairs. Hence, every causal spike pair is followed by an anti-causal pair, and it follows directly that  $\int_0^\infty p^+(t) dt = \int_0^\infty \hat{p}^-(t) dt$ . Using the definition of the deviation eq. (20), we thus note that

$$\int_0^\infty \delta(t) dt = 0. \quad (21)$$

To prove the existence of fixed points of SAL, we turn to the expectation value for a weight update  $\langle \dot{W}_{ij} \rangle$ . The expectation value for a symmetric STDP kernel is given by

$$\langle \dot{W}_{ij} \rangle = \int_{-\infty}^0 f^-(t) p^-(t) dt + \int_0^\infty f^+(t) p^+(t) dt \quad (22)$$

$$= \int_0^\infty f^+(t) [p^+(t) - \hat{p}^-(t)] dt. \quad (23)$$

For brevity, we omit the synapse indices here for the STDD  $p_{ij}$ . As demonstrated in fig. 7, the STDD is symmetric ( $p_0^+ = \hat{p}_0^-$ ) for  $W_{ij} = W_{ji}$  if and only if the PSP shape is rectangular, independent of the choice of the biases, and for arbitrary PSPs if  $b_1 = b_2$ . Hence, the average weight update is zero,  $\langle \dot{W}_{ij} \rangle = 0$ , and  $W_{12} = W_{21}$  is a unique fixed point.

We now show that the fixed points are stable under small perturbations of one weight, which is represented by  $p_\varepsilon$ . That means that  $\langle \dot{W}_{21} \rangle < 0$  if  $\varepsilon > 0$ , i.e.,  $W_{21} > W_{12}$  and vice versa that  $\langle \dot{W}_{21} \rangle > 0$  if  $\varepsilon < 0$ . Since the STDDs are identical (just time-mirrored) for the two synapses, the same derivation holds for  $W_{12}$ . We start with  $\varepsilon > 0$ . The expectation value analog to eq. (22) is

$$\langle \dot{W}_{21} \rangle = \int_0^\infty f^+(t) [p_\varepsilon^+(t) - \hat{p}_\varepsilon^-(t)] dt \quad (24)$$

$$= \int_0^\infty f^+(t) [p_0^+(t) + \delta(t) - \hat{p}_0^-(t)] dt \quad (25)$$

$$= \int_0^\infty f^+(t) \delta(t) dt. \quad (26)$$

By introducing the variable  $\Delta(t) := \int_0^t \delta(t') dt'$  and integrating by parts, we can reformulate eq. (24) as

$$\langle \dot{W}_{21} \rangle = f^+(t) \Delta(t) \Big|_0^\infty - \int_0^\infty f^{+'}(t) \Delta(t) dt. \quad (27)$$

Because of  $\Delta(0) = 0$  and  $\lim_{t \rightarrow \infty} \Delta(t) = 0$  (see fig. 8), the boundary terms vanish. Importantly, because of  $\Delta(t) > 0$  for all  $t > 0$  and because of our assumption  $f^{+'}(t) > 0$ , the integral evaluates to a finite positive number. Therefore, for a positive  $\varepsilon$ , we have  $\langle \dot{W}_{21} \rangle < 0$ . For a negative  $\varepsilon$ ,  $\delta$  and consequently  $\Delta$  change the sign and therefore the integral term evaluates to a negative number, and  $\langle \dot{W}_{21} \rangle < 0$ . Hence, in the vicinity around the fixed point  $W_{21} = W_{12}$ , the weight updates move  $W_{21}$  towards  $W_{12}$ , which shows that  $W_{12} = W_{21}$  is a stable solution.

#### 4.3 Sampling with spikes

To demonstrate the effectiveness of SAL in networks with strong recurrence, we turn to spiking sampling networks (SSNs), which we train to represent arbitrary Boltzmann distributions. The framework of SSNs was introduced in [61]. It builds on the theory of Boltzmann machines (BMs), which originate from the field of statistical physics and are used to represent probability distributions over binary variables  $z_i$  of a system of  $N$  coupled spins or – in the language of neural networks – a recurrent network of stochastic binary neurons.

For this system, we define an energy function

$$E(\mathbf{z}) = -\frac{1}{2} \mathbf{z}^\top \mathbf{W} \mathbf{z} - \mathbf{b}^\top \mathbf{z}, \quad (28)$$

that assigns a scalar energy value to every possible configuration  $\mathbf{z} \in [0, 1]^N$  of the system.  $\mathbf{W} \in \mathbb{R}^{N \times N}$  is a symmetric weight matrix with vanishing diagonal that contains the interactions strengths between the neurons and  $\mathbf{b} \in \mathbb{R}^N$  the vector of neuronal biases.

The probability of a state  $\mathbf{z}$  is determined by the Boltzmann distribution,

$$P(\mathbf{z}) = \frac{1}{Z} \exp(-E(\mathbf{z})), \quad (29)$$

where  $Z = \sum_{\mathbf{z}} \exp(-E(\mathbf{z}))$  is the partition function.

To make the link between the theory of BMs and spiking neural networks, we map the refractoriness of a neuron to the binary variable  $z_i$ , where  $z_i = 1$  is assigned to a neuron which is refractory and  $z_i = 0$  to a neuron which is not. The neuron model employed here is the same as defined in section 4.1 by eqs. (7) to (10). Driven by its intrinsic randomness, the network randomly evolves over time and explores different network states along some random trajectory through the state space, which we call sampling. The relative frequency by which the states  $\mathbf{z}$  are visited represents the underlying joint probability distribution  $p(\mathbf{z})$ . In [61], the authors have shown the equivalence between Markov chain Monte Carlo (MCMC) from Boltzmann distributions and the sampling with networks of spiking GLMs.

By choosing a suitable number of neurons and partitioning the network into a population of *visible* and *hidden* neurons, the marginal distribution over the visible population can approximate any joint probability distribution.

The training of the network consists of minimizing the difference between the network distribution  $p(\mathbf{z})$  and the target distribution  $p^*(\mathbf{z})$ . To do so, we minimize the Kullback-Leibler divergence ( $D_{\text{KL}}$ ), a standard measure of similarity between two probability distributions. Minimization with respect to the network’s weights and biases yields optimized parameters

$$\hat{\theta} = \arg \min_{\theta} D_{\text{KL}}(p||p^*), \quad \theta \in \{\mathbf{b}, \mathbf{W}\}, \quad (30)$$

where  $D_{\text{KL}}(p||p^*) := \sum_{\mathbf{z}} p(\mathbf{z}) \log \frac{p(\mathbf{z})}{p^*(\mathbf{z})}$ . The weights and biases are updated iteratively by performing gradient descent on  $D_{\text{KL}}$ . Training is split into two reoccurring phases: In the *wake phase*, the network is constrained by the target distribution (it “sees” the target pattern), while during the *sleep phase*, it is allowed to sample freely from its internal distribution. This results in the wake-sleep algorithm used for training BMs [3],

$$\Delta W_{ij} = \eta_W [\langle z_i z_j \rangle_{\text{wake}} - \langle z_i z_j \rangle_{\text{sleep}}] \quad (31)$$

$$\Delta b_i = \eta_b [\langle z_i \rangle_{\text{wake}} - \langle z_i \rangle_{\text{sleep}}], \quad (32)$$

where  $\langle \cdot \rangle_{\text{wake}} = \langle \cdot \rangle_{p^*(\mathbf{z})}$  denotes the expectation value with respect to the target distribution  $p^*(\mathbf{z})$  and  $\langle \cdot \rangle_{\text{sleep}} = \langle \cdot \rangle_{p(\mathbf{z}|\mathbf{b}, \mathbf{W})}$  with respect to the distribution of the free model. We refer to eqs. (31) and (32) as *state-based* wake-sleep.

In an SSN, the estimation of the correlations  $\langle z_i z_j \rangle_x$  can be realized directly on the spike-trains by a standard STDP-rule as introduced in eq. (11), which we refer to as *spike-based* wake-sleep. The STDP-window has the shape of a left-right symmetric triangle with  $f(\Delta t) = \max\left(-\frac{\Delta t}{\tau_{\text{ref}}}, -1, 0\right)$  and  $\alpha_{a/c} = 1$  for the wake and  $\alpha_{a/c} = -1$  for the sleep phase. In the limit of small learning rates, the accumulated weight updates using spike-based STDP after the two phases is equivalent to the weight update produced by state-based wake-sleep. This allows us to minimize  $D_{\text{KL}}$  in an efficient online manner, relying only on the knowledge of last spike times instead of accumulated expectation values.

	synaptic noise		plasticity noise	
	w/o SAL	w/ SAL	w/o SAL	w/ SAL
$\mathbf{W}^*$			$\mathcal{U}[-1, 1]$	
$\mathbf{b}^*$			$\mathcal{U}[-1, 1]$	
$\mathbf{W}_{\text{init}}$			$\mathcal{N}(0, 0.2)$	
$\mathbf{b}_{\text{init}}$			$\mathcal{N}(0, 0.2)$	
$\sigma_{\text{init}}^{\text{noise}}$	variable	variable	0.2	0.2
$\sigma_{\text{STDP}}^{\text{noise}}$	0	0	variable	variable
$\eta_W$	0.01	0.01	0.001	0.001
$\eta_b$	0.01	0.01	0.001	0.001
$\eta_{\text{SAL}}$	0	0.01	0	0.002
$\tau_{\text{ref}}$		50 timesteps		
duration		1000 $\tau_{\text{ref}}$		
sleep phase				

**Table 1:** Simulation parameters for the SSN experiments.

### 4.3.1 Simulation details

We train a fully connected BM of size  $N = 7$  to sample from random target distributions of the same dimension. To compute the target distributions  $p^*$ , we generate Boltzmann distributions using eq. (29) with parameters  $\mathbf{W}^*$  and  $\mathbf{b}^*$  sampled from a uniform distribution. This ensures that the BM is able to solve the task of representing  $p^*$  with high precision.

Prior to training, the network is initialized with random biases  $\mathbf{b}_{\text{init}}$  and weights  $\mathbf{W}_{\text{init}}$  drawn from a normal distribution. The upper triangle of  $\mathbf{W}_{\text{init}}$  is copied to the lower triangle to ensure symmetry.

Depending on the experiment type, parameter noise is added to the network upon initialization: In the *synaptic noise scenario*, Gaussian noise with  $\mu = 0$  and a standard deviation  $\sigma_{\text{init}}^{\text{noise}}$  is added to each weight to model weight asymmetry of different strength prior to training. In the *plasticity noise scenario*, all runs are conducted with Gaussian noise with  $\sigma_{\text{init}}^{\text{noise}} = 0.2$  added to  $\mathbf{W}_{\text{init}}$ . Additionally, we also add noise to the STDP factors: For each synapse  $W_{ij}$ , a random noise value  $\xi_{ij}$  is drawn from a normal distribution with  $\mu = 0$  and  $\sigma_{\text{STDP}}^{\text{noise}}$ . For the wake phase, this is added to both causal and anticausal prefactors,  $\alpha_{a/c} + \xi_{ij}$ , and subtracted for the sleep phase,  $\alpha_{a/c} - \xi_{ij}$ . This models synaptic heterogeneity and ensures that reciprocal synapses produce non-equal weights updates, although they receive the same spike trains and should produce as the same updates as expected.

Each scenario is simulated twice, one time without SAL and a second time with an additional SAL phase. To accelerate learning, sampling during the wake phase is replaced by calculated target coactivation  $\langle z_i z_j \rangle_{\text{wake}}$  and rates  $\langle z_i \rangle_{\text{wake}}$ . Weight updates are applied in batches after each phase (wake-sleep or SAL). Learning rates are optimized through visual inspection, and training progress is monitored using the  $D_{\text{KL}}$  between  $p$  and  $p^*$ . For each scenario, a reference run without the specific noise type is conducted (blue/purple lines and markers in figs. 4 and 5), and training is halted when the  $D_{\text{KL}}$  ceases to decrease. For each noise level in figs. 4 and 5, we train the same 5 distributions with 4 seeds each, resulting in different 20 runs, and compare the  $D_{\text{KL}}$  and variance of weight differences  $\text{Var}(W_{ij} - W_{ji})$  between runs with and without SAL. The relevant simulation parameters are given in table 1.



## 4.4 Spiking cortical microcircuits

To demonstrate SAL in the context of biologically plausible backpropagation (BP) in spiking networks, we turn to the model of cortical microcircuits presented in [29] and extend it to communication with actual spikes instead of rates.

### 4.4.1 Mathematical description of the model

The general scheme of the circuit is depicted in fig. 6. For a proof of the equivalence between the (rate-based) microcircuit model and BP, we refer the reader to [29] and [41].

In the absence of a teaching signal, e.g., from higher cortical areas, pyramidal cells function as representation units, responsible for feed-forward activation. We denote variables associated with these cells with a superscript  $\cdot^P$ . They receive bottom-up sensory inputs via their basal dendrites and top-down error/learning signals through the apical dendrite, integrating these inputs at the soma. Such preferential targeting of dendritic compartments by top-down/bottom-up projections is consistent with observations of pyramidal cells in cortex [87–91]. In the model, the complex dynamics of pyramidal cells are modelled with a simplified three-compartment model, which includes distinct basal, apical, and somatic voltages.

Interneurons (indicated by a superscript  $\cdot^I$ ) are located in the hidden layers of the network and aim to replicate the activation of pyramidal cells in the layer above. They are divided into two compartments, representing the dendritic tree and the soma. Across the layers, the populations of pyramidal neurons and interneurons are organized such that the number of interneurons in the hidden layers matches the number of pyramidal cells in the layer above. Pyramidal cells project laterally to these interneurons and receive signal back from them to their apical tree.

In accordance with the previous models of this work (sections 2.4.1 and 4.1), neurons are modelled as GLMs, i.e., we use the spiking mechanism presented in eq. (9) together with the activation function eq. (10). The membrane potential dynamics of pyramidal cells integrate the compartments,

$$\mathbf{u}_\ell^P(t) = \mathbf{b}_\ell + \lambda^{\text{bas}} \mathbf{v}_\ell^{\text{bas}}(t) + \lambda^{\text{api}} \mathbf{v}_\ell^{\text{api}}(t), \quad (33)$$

where  $\mathbf{b}_\ell$  represents the neuronal biases and  $\mathbf{v}_\ell^{\text{bas}}$  and  $\mathbf{v}_\ell^{\text{api}}$  the basal and apical voltages respectively together with their coupling strengths  $\lambda^{\text{bas}}$  and  $\lambda^{\text{api}}$ .

The basal compartment  $\mathbf{v}_\ell^{\text{bas}}(t) = \mathbf{W}_{\ell,\ell-1}^{\text{PP}} \hat{\mathbf{r}}_{\ell-1}^P(t)$  receives input spikes from the pyramidal neurons in the layer below. Here,  $\mathbf{W}_{\ell,\ell-1}^{\text{PP}}$  denotes the bottom-up weights from layer  $\ell-1$  to  $\ell$  and  $\hat{\mathbf{r}}_{\ell-1,i}^P(t) = \int_0^\infty \kappa(s) S_{\ell-1,i}^r(t-s) ds$  the filtered spike train from neuron  $i$  in layer  $\ell-1$ .

The apical compartment  $\mathbf{v}_\ell^{\text{api}}(t) = \mathbf{B}_{\ell,\ell+1}^{\text{PP}} \hat{\mathbf{r}}_{\ell+1}^P(t) + \mathbf{L}_{\ell,\ell}^{\text{PI}} \hat{\mathbf{r}}_\ell^I(t)$  integrates the top-down input from the upper layer (through  $\mathbf{B}_{\ell,\ell+1}^{\text{PP}}$ ) and compares it to the activity of the interneurons received through the lateral weights  $\mathbf{L}_{\ell,\ell}^{\text{PI}}$ .

In this model, interneurons consist of two compartments, representing the soma and a dendritic tree. Their somatic voltage is given by

$$\mathbf{u}_\ell^I(t) = \mathbf{b}_\ell + \lambda^{\text{den}} \mathbf{v}_\ell^{\text{den}}(t) = \mathbf{b}_\ell + \lambda^{\text{den}} \mathbf{L}_{\ell,\ell}^{\text{IP}} \hat{\mathbf{r}}_\ell^P(t), \quad (34)$$

receiving input from the population of pyramidal neurons in the same layer through afferent lateral weights  $\mathbf{L}_{\ell,\ell}^{\text{IP}}$ .

During training, the top layer neurons are described by

$$\mathbf{u}_L^P(t) = \mathbf{b}_L + \lambda^{\text{bas}} \mathbf{v}_L^{\text{bas}}(t) + \lambda^{\text{nudge}} \mathbf{v}_L^{\text{api}}(t), \quad (35)$$

where the apical compartment induces a nudging by the current error signal, i.e. the difference between the target voltage  $\mathbf{u}^{\text{tgt}}$  and bottom-up input plus bias,  $\mathbf{v}_L^{\text{api}} = \mathbf{u}^{\text{tgt}} - (\mathbf{b}_L + \lambda^{\text{bas}} \bar{\mathbf{v}}_L^{\text{bas}})$ . Note that we have introduced a time-smoothed version  $\bar{\mathbf{v}}_L^{\text{bas}}$  of the basal input, which we obtain by taking a moving average over present and past voltages  $\mathbf{v}_L^{\text{bas}}(t)$ . The averaging of  $\mathbf{v}^{\text{bas}}$  is needed because the target is provided by a vector with smooth, continuous values and hence the error signal encoded in  $\mathbf{v}^{\text{api}}$  is required to be smooth in time as well.  $\lambda^{\text{nudge}}$  controls the nudging strength of the target, which is set to zero in absence of a teaching signal.

The microcircuit model has to be operated in the so-called *self-predicting state*, in which  $\mathbf{u}_\ell^I$  matches  $\mathbf{u}_\ell^P$  in the absence of a top-down teaching signal. In this state, the apical voltage  $\mathbf{v}_\ell^{\text{api}}$  is zero when the network receives no training signal (or the network has learned perfectly) because the top-down input from the layer  $\ell+1$  and the lateral input from the interneuron in layer  $\ell$  cancel. The self-predicting state is realized if  $\mathbf{L}_{\ell,\ell}^{\text{IP}} = \frac{\lambda^{\text{bas}}}{\lambda^{\text{den}}} \mathbf{W}_{\ell+1,\ell}^{\text{PP}}$ , which can be dynamically learned as done in the original model. Here, we set it for computational efficiency.

Following this logic,  $\mathbf{L}_{\ell,\ell}^{\text{PI}}$  and  $\mathbf{B}_{\ell,\ell+1}^{\text{PP}}$  are matched such that  $\mathbf{v}_\ell^{\text{api}} = 0$  in absence of a top-down teaching signal. This, too, can be achieved dynamically by a local learning rule [29]; here, we also set  $\mathbf{L}_{\ell,\ell}^{\text{PI}} = -\mathbf{B}_{\ell,\ell+1}^{\text{PP}}$ .

Following the derivation presented in [29], one can show that  $\mathbf{v}_\ell^{\text{api}}$  effectively encodes a local error signal: If  $\mathbf{u}_{\ell+1}^P$  is nudged towards a target, a residual voltage is induced in  $\mathbf{v}_\ell^{\text{api}}$ . It can be shown that this model together with a local plasticity rule inspired by [92] can approximate the BP algorithm [29, 41]. Our spiking adaptation of the bottom-up learning rule connecting neuron  $i$  to neuron  $j$  is

$$\begin{aligned} \dot{W}_{\ell,\ell-1,j}^{\text{PP}}(t) = & \quad (36) \\ \eta \left[ \varphi(\bar{u}_{\ell,j}^P(t)) - \varphi(b_{\ell,j} + \lambda^{\text{bas}} \bar{v}_{\ell,j}^{\text{bas}}(t)) \right] S_{\ell-1,i}^P(t), \end{aligned}$$

where  $\eta$  is the learning rate and  $\varphi(u) = \tau_{\text{ref}}^{-1} \exp(u)$  the activation function of the GLM. In addition to  $\bar{v}^{\text{bas}}$ , the learning rule also requires the smoothed somatic membrane potential  $\bar{u}^P$ .

Similar to phaseless alignment learning (PAL) for the rate-based case [41], we endow the (now spiking) microcircuit model with SAL, replacing feedback alignment (FA) with dynamical alignment of backward connections  $\mathbf{B}^{\text{PP}}$  with their feed-forward partner weights  $\mathbf{W}^{\text{PP}}$ .

### 4.4.2 Simulation details

Using this setup, we demonstrate that SAL outperforms FA, and retains the efficient credit assignment of BP. We point out that the simulations are carried out with fully recurrent dynamics and fully spike-based communication as described by eqs. (33) to (36), setting our work apart from similar approaches, where spikes are replaced by rates and neuronal dynamics by steady-state approximations, or the challenges of recurrence are lifted by computing the forward and backward passes separately.

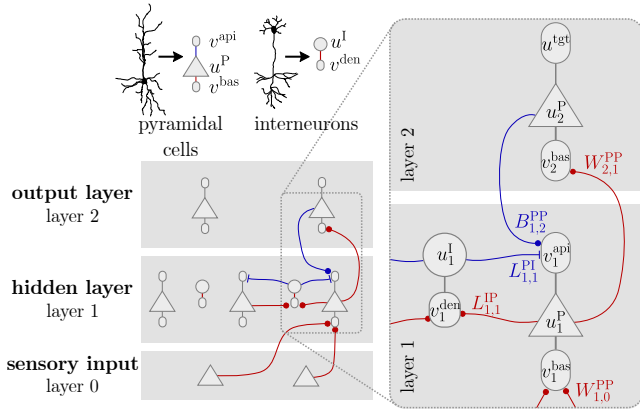
We use a teacher microcircuit network to produce a non-linear input-output mapping that the student network has to learn. The teacher consists of one hidden microcircuit and a pyramidal output neuron. Teacher and student have the same size and parametrization, but only the student is

nudged by the target. Crucially, to achieve the optimal output, the student network has to learn the exact same weights as the teacher, which is only possible if a meaningful error signal arrives at the hidden neuron.

Training is divided into epochs of 20 shuffled training inputs, an optional SAL phase of five inputs, and a validation phase without nudging of six validation inputs. Each input consists of a voltage that is converted into spike trains using eqs. (9) and (10). The input for the training phases consists of nine equally spaced voltages between  $-3$  and  $3$ ; the input for the validation consists of six voltages between  $-2.7$  and  $2.7$ . In all cases, the targets are the time-averaged somatic voltages  $u_L^P$  recorded from the output neuron of the teacher network.

When using “BP”, the value of  $W_{2,1}^{PP}$  is copied to  $B_{1,2}^{PP}$  after every time step. When using FA, When using SAL,  $\lambda^{\text{api}}$  is set to 1 during the SAL-phases. Otherwise, SAL-plasticity is not activated in  $B_{1,2}^{PP}$ .

All relevant simulation parameters are given in table 2.



**Figure 9:** Schematic of the microcircuit model: bio-plausible transportation of error signals and local error representation in apical dendrites. Adapted from [41]

#### 4.5 Other PSP shapes

For calculating the *average relative deviation* between the true attractor and the diagonal  $W_{21} = W_{12}$  in the phase plane diagram fig. 7, we define a map  $W'_{21} \rightarrow W'_{12} = g(W'_{21})$ , that characterizes all points  $W' = (W'_{21}, W'_{12})$  on the attractor. The function  $g$  does not need to be defined or fitted by an analytical function, instead, we determine enough points on the attractors numerically. It also does not matter if we map  $W'_{21} \rightarrow W'_{12}$  or vice versa, since the problem is symmetric under the exchange of indices. We define the relative deviation of a point on the attractor from the diagonal by

$$\delta(W_{21}) := \frac{W_{21} - g(W_{21})}{W_{21} + g(W_{21})}. \quad (37)$$

The average relative deviation is then

$$D := \frac{1}{W_{21}^{\max} - W_{21}^{\min}} \int_{W_{21}^{\min}}^{W_{21}^{\max}} |\delta(W'_{21})| dW'_{21}, \quad (38)$$

where  $W_{21}^{\max}$  and  $W_{21}^{\min}$  are the maximal and minimal weight values in the phase plane diagram.

	BP	FA	SAL
$\lambda^{\text{api}}$		0.02	
$\lambda^{\text{nudge}}$		0.6	
$\lambda^{\text{bas}}$		1.0	
$\lambda^{\text{den}}$		1.0	
$t_{\text{moving average}}$		2000 timesteps	
$\tau_{\text{ref}}$		10 timesteps	
$\eta$ for $W_{1,0}^{PP}$	0.2	0.2	0.2
$\eta$ for $W_{2,1}^{PP}$	0.003	0.003	0.003
$\eta$ for $B_{1,2}^{PP}$	n.a.	0.0	0.001
$b_1$		-1.0	
$b_2$		-1.0	
teacher $W_{1,0}^{PP}$		2.0	
teacher $W_{2,1}^{PP}$		2.0	
training values		20	
per iteration			
validation values		6	
per iteration			
values for SAL	0	0	5
per iteration			
presentation time		2000 $\tau_{\text{ref}}$	
student init $W_{1,0}^{PP}$		$\mathcal{U}[-3, 3]$	
student init $W_{2,1}^{PP}$		$\mathcal{U}[-3, 3]$	
student init $B_{1,2}^{PP}$	$= W_{2,1}^{PP}$	$\mathcal{U}[-3, 3]$	$\mathcal{U}[-3, 3]$

**Table 2:** Simulation parameters for the cortical microcircuits.

#### Code availability

The simulations were performed by custom code written in Python (v3.11), numpy (v2.0) and numba (v0.60). All code is made available under <https://github.com/unibe-cns/sal-code>.

#### Acknowledgements

We would like to thank Everton Agnes for an insightful discussion about the plethora of plasticity mechanisms observed in cortex. This research has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement Nos. 720270, 785907, and 945539 (Human Brain Project, HBP) and from the European Union’s Horizon Europe Programme under the Specific Grant Agreement No. 101147319 (EBRAINS 2.0 Project). Some simulations were performed on the bwFor-Cluster NEMO, supported by the state of Baden-Württemberg through bwHPC and the German Research Foundation (DFG) through grant no. INST 39/963-1 FUGG. We further acknowledge the use of EBRAINS and Fenix Infrastructure resources, which are partially funded from the European Union’s Horizon 2020 research and innovation programme through the ICEI project under the grant agreement No. 800858. The contribution of KM has received funding by the Volkswagen Foundation and the Swiss National Science Foundation. Finally, we would like to express our deepest gratitude to the Manfred Stärk Foundation, whose unwavering support has made this, and many other projects possible.

## References

1. Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences* **79**, 2554–2558 (1982).
2. Hinton, G. E. & Sejnowski, T. J. *Optimal perceptual inference* in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* **448** (1983), 448–453.
3. Hinton, G. E., Dayan, P., Frey, B. J. & Neal, R. M. The “wake-sleep” algorithm for unsupervised neural networks. *Science* **268**, 1158–1161 (1995).
4. Linnainmaa, S. Taylor expansion of the accumulated rounding error. *BIT Numerical Mathematics* **16**, 146–160 (1976).
5. Werbos, P. J. *Applications of advances in nonlinear sensitivity analysis* in *System Modeling and Optimization: Proceedings of the 10th IFIP Conference New York City, USA, August 31–September 4, 1981* (2005), 762–770.
6. Rumelhart, D. E., Hinton, G. E. & Williams, R. J. Learning representations by back-propagating errors. *nature* **323**, 533–536 (1986).
7. Whittington, J. C. & Bogacz, R. Theories of error backpropagation in the brain. *Trends in cognitive sciences* **23**, 235–250 (2019).
8. Richards, B. A. *et al.* A deep learning framework for neuroscience. *Nature neuroscience* **22**, 1761–1770 (2019).
9. Lillicrap, T. P., Santoro, A., Marris, L., Akerman, C. J. & Hinton, G. Backpropagation and the brain. *Nature Reviews Neuroscience* **21**, 335–346 (2020).
10. Marder, E. & Goaillard, J.-M. Variability, compensation and homeostasis in neuron and network function. *Nature Reviews Neuroscience* **7**, 563–574 (2006).
11. Watt, A. J. & Desai, N. S. Homeostatic plasticity and STDP: keeping a neuron’s cool in a fluctuating world. *Frontiers in synaptic neuroscience* **2**, 1486 (2010).
12. Abbott, L. F. & Nelson, S. B. Synaptic plasticity: taming the beast. *Nature neuroscience* **3**, 1178–1183 (2000).
13. Davis, G. W. & Bezprozvanny, I. Maintaining the stability of neural function: a homeostatic hypothesis. *Annual review of physiology* **63**, 847–869 (2001).
14. Lee, H.-K. & Kirkwood, A. Mechanisms of homeostatic synaptic plasticity in vivo. *Frontiers in Cellular Neuroscience* **13**, 520 (2019).
15. Yang, J. & Prescott, S. A. Homeostatic regulation of neuronal function: importance of degeneracy and pleiotropy. *Frontiers in Cellular Neuroscience* **17**, 1184563 (2023).
16. Berkes, P., Orbán, G., Lengyel, M. & Fiser, J. Spontaneous cortical activity reveals hallmarks of an optimal internal model of the environment. *Science* **331**, 83–87 (2011).
17. Haefner, R. M., Berkes, P. & Fiser, J. Perceptual decision-making as probabilistic inference by neural sampling. *Neuron* **90**, 649–660 (2016).
18. Orbán, G., Berkes, P., Fiser, J. & Lengyel, M. Neural variability and sampling-based probabilistic representations in the visual cortex. *Neuron* **92**, 530–543 (2016).
19. Kungl, A. F. *et al.* Accelerated physical emulation of bayesian inference in spiking neural networks. *Frontiers in neuroscience* **13**, 1201 (2019).
20. Billaudelle, S. *et al.* *Versatile emulation of spiking neural networks on an accelerated neuromorphic substrate* in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)* (2020), 1–5.
21. Czischek, S. *et al.* Spiking neuromorphic chip learns entangled quantum states. *SciPost Physics* **12**, 039 (2022).
22. Klassert, R., Baumbach, A., Petrovici, M. A. & Gärtner, M. Variational learning of quantum ground states on spiking neuromorphic hardware. *Iscience* **25** (2022).
23. Hinton, G. E. Training products of experts by minimizing contrastive divergence. *Neural computation* **14**, 1771–1800 (2002).
24. Crick, F. The recent excitement about neural networks. *Nature* **337**, 129–132 (1989).
25. Lansdell, B. J., Prakash, P. R. & Kording, K. P. Learning to solve the credit assignment problem. *arXiv preprint arXiv:1906.00889* (2019).
26. Ernoult, M. M. *et al.* *Towards scaling difference target propagation by learning backprop targets* in *International Conference on Machine Learning* (2022), 5968–5987.
27. Xie, X. & Seung, H. S. Equivalence of backpropagation and contrastive Hebbian learning in a layered network. *Neural computation* **15**, 441–454 (2003).
28. Scellier, B. & Bengio, Y. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in computational neuroscience* **11**, 24 (2017).
29. Sacramento, J., Ponte Costa, R., Bengio, Y. & Senn, W. Dendritic cortical microcircuits approximate the backpropagation algorithm. *Advances in neural information processing systems* **31** (2018).
30. Pozzi, I., Bohte, S. & Roelfsema, P. Attention-Gated Brain Propagation: How the brain can implement reward-based error backpropagation. *Advances in neural information processing systems* **33**, 2516–2526 (2020).
31. Kolen, J. F. & Pollack, J. B. *Backpropagation without weight transport* in *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN’94)* **3** (1994), 1375–1380.
32. Payeur, A., Guerguiev, J., Zenke, F., Richards, B. A. & Naud, R. Burst-dependent synaptic plasticity can coordinate learning in hierarchical circuits. *Nature neuroscience* **24**, 1010–1019 (2021).
33. Roelfsema, P. R. & Ooyen, A. v. Attention-gated reinforcement learning of internal representations for classification. *Neural computation* **17**, 2176–2214 (2005).
34. Pozzi, I., Bohté, S. & Roelfsema, P. A biologically plausible learning rule for deep learning in the brain. *arXiv preprint arXiv:1811.01768* (2018).
35. Haider, P. *et al.* Latent equilibrium: A unified learning theory for arbitrarily fast computation with arbitrarily slow neurons. *Advances in neural information processing systems* **34**, 17839–17851 (2021).
36. Ellenberger, B. *et al.* Backpropagation through space, time, and the brain. *arXiv preprint arXiv:2403.16933* (2024).
37. Senn, W. *et al.* A neuronal least-action principle for real-time learning in cortical circuits. *ELife* **12**, RP89674 (2024).
38. Lillicrap, T. P., Cownden, D., Tweed, D. B. & Akerman, C. J. Random synaptic feedback weights support error backpropagation for deep learning. *Nature communications* **7**, 13276 (2016).
39. Bartunov, S. *et al.* Assessing the scalability of biologically-motivated deep learning algorithms and architectures. *Advances in neural information processing systems* **31** (2018).
40. Moskovitz, T. H., Litwin-Kumar, A. & Abbott, L. Feedback alignment in deep convolutional networks. *arXiv preprint arXiv:1812.06488* (2018).
41. Max, K. *et al.* Learning efficient backprojections across cortical hierarchies in real time. *Nature Machine Intelligence*, 1–12 (2024).
42. Burbank, K. S. Mirrored STDP implements autoencoder learning in a network of spiking neurons. *PLoS computational biology* **11**, e1004566 (2015).
43. Petrovici, M. A. *et al.* Characterization and compensation of network-level anomalies in mixed-signal neuromorphic modeling platforms. *PloS one* **9**, e108590 (2014).
44. Pehle, C. *et al.* The BrainScaleS-2 accelerated neuromorphic system with hybrid plasticity. *Frontiers in Neuroscience* **16**, 795876 (2022).

45. Grübl, A., Billaudelle, S., Cramer, B., Karasenko, V. & Schemmel, J. Verification and design methods for the brain-scales neuromorphic hardware system. *Journal of Signal Processing Systems* **92**, 1277–1292 (2020).
46. Kawaguchi, Y. Groupings of nonpyramidal and pyramidal cells with specific physiological and morphological characteristics in rat frontal cortex. *Journal of neurophysiology* **69**, 416–431 (1993).
47. Liu, Y. *et al.* Neuronal diversity and stereotypy at multiple scales through whole brain morphometry. *Nature Communications* **15**, 10269 (2024).
48. Mednikova, Y. S., Rogal, A. V., *et al.* Heterogeneity of the Neural Composition of Cortical Regions as a Condition for a Wide Range Regulating Spontaneous Activity. *Journal of Behavioral and Brain Science* **10**, 220 (2020).
49. Rathour, R. K. & Kaphzan, H. Voltage-gated ion channels and the variability in information transfer. *Frontiers in Cellular Neuroscience* **16**, 906313 (2022).
50. Faisal, A. A., Selen, L. P. & Wolpert, D. M. Noise in the nervous system. *Nature reviews neuroscience* **9**, 292–303 (2008).
51. Brunel, N. Dynamics of sparsely connected networks of excitatory and inhibitory spiking neurons. *Journal of computational neuroscience* **8**, 183–208 (2000).
52. Fourcaud, N. & Brunel, N. Dynamics of the firing probability of noisy integrate-and-fire neurons. *Neural computation* **14**, 2057–2110 (2002).
53. Jordan, J. *et al.* Deterministic networks for probabilistic computing. *Scientific reports* **9**, 18303 (2019).
54. Dold, D. *et al.* Stochasticity from function—why the bayesian brain may need no noise. *Neural networks* **119**, 200–213 (2019).
55. Müller, E. *et al.* Extending brainscales OS for BrainScaleS-2. *arXiv preprint arXiv:2003.13750* (2020).
56. Koch, G., Ponzio, V., Di Lorenzo, F., Caltagirone, C. & Veniero, D. Hebbian and anti-Hebbian spike-timing-dependent plasticity of human cortico-cortical connections. *Journal of Neuroscience* **33**, 9725–9733 (2013).
57. Pfeil, T. *et al.* Six networks on a universal neuromorphic computing substrate. *Frontiers in neuroscience* **7**, 11 (2013).
58. Serrano-Gotarredona, T., Masquelier, T., Prodromakis, T., Indiveri, G. & Linares-Barranco, B. STDP and STDP variations with memristors for spiking neuromorphic learning systems. *Frontiers in neuroscience* **7**, 2 (2013).
59. Qiao, N. *et al.* A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128K synapses. *Frontiers in neuroscience* **9**, 141 (2015).
60. Davies, M. *et al.* Loihi: A neuromorphic manycore processor with on-chip learning. *Ieee Micro* **38**, 82–99 (2018).
61. Buesing, L., Bill, J., Nessler, B. & Maass, W. Neural dynamics as sampling: a model for stochastic computation in recurrent networks of spiking neurons. *PLoS computational biology* **7**, e1002211 (2011).
62. Petrovici, M. A., Bill, J., Bytschok, I., Schemmel, J. & Meier, K. Stochastic inference with spiking neurons in the high-conductance state. *Physical Review E* **94**, 042312 (2016).
63. Pouget, A., Beck, J. M., Ma, W. J. & Latham, P. E. Probabilistic brains: knowns and unknowns. *Nature neuroscience* **16**, 1170–1178 (2013).
64. Fiser, J., Berkes, P., Orbán, G. & Lengyel, M. Statistically optimal perception and learning: from behavior to neural representations. *Trends in cognitive sciences* **14**, 119–130 (2010).
65. Neftci, E., Das, S., Pedroni, B., Kreuz-Delgado, K. & Cauwenberghs, G. Event-driven contrastive divergence for spiking neuromorphic systems. *Frontiers in neuroscience* **7**, 272 (2014).
66. Huang, Y. & Rao, R. P. Predictive coding. *Wiley Interdisciplinary Reviews: Cognitive Science* **2**, 580–593 (2011).
67. Millidge, B., Seth, A. & Buckley, C. L. Predictive coding: a theoretical and experimental review. *arXiv preprint arXiv:2107.12979* (2021).
68. Mikulasch, F. A., Rudelt, L., Wibral, M. & Priesemann, V. Where is the error? Hierarchical predictive coding through dendritic error computation. *Trends in Neurosciences* **46**, 45–59 (2023).
69. Nøkland, A. Direct feedback alignment provides learning in deep neural networks. *Advances in neural information processing systems* **29** (2016).
70. Liao, Q., Leibo, J. & Poggio, T. How important is weight symmetry in backpropagation? in *Proceedings of the AAAI Conference on Artificial Intelligence* **30** (2016).
71. Bengio, Y. How auto-encoders could provide credit assignment in deep networks via target propagation. *arXiv preprint arXiv:1407.7906* (2014).
72. Lee, D.-H., Zhang, S., Fischer, A. & Bengio, Y. Difference target propagation in *Joint european conference on machine learning and knowledge discovery in databases* (2015), 498–515.
73. Meulemans, A., Carzaniga, F., Suykens, J., Sacramento, J. & Grewe, B. F. A theoretical framework for target propagation. *Advances in Neural Information Processing Systems* **33**, 20024–20036 (2020).
74. Meulemans, A. *et al.* Credit assignment in neural networks through deep feedback control. *Advances in Neural Information Processing Systems* **34** (2021).
75. Akrou, M., Wilson, C., Humphreys, P. C., Lillicrap, T. & Tweed, D. Deep learning without weight transport. *arXiv preprint arXiv:1904.05391* (2019).
76. O’Connor, P., Gavves, E. & Welling, M. Training a spiking neural network with equilibrium propagation in *The 22nd international conference on artificial intelligence and statistics* (2019), 1516–1523.
77. Boikov, I. K., de Rossi, A. & Petrovici, M. A. Ultrafast neural sampling with spiking nanolasers. *arXiv preprint arXiv:2501.14446* (2025).
78. Schemmel, J. *et al.* A wafer-scale neuromorphic hardware system for large-scale neural modeling in *2010 IEEE International Symposium on Circuits and Systems (ISCAS)* (2010), 1947–1950.
79. Moradi, S., Qiao, N., Stefanini, F. & Indiveri, G. A scalable multicore architecture with heterogeneous memory structures for dynamic neuromorphic asynchronous processors (DYNAPs). *IEEE transactions on biomedical circuits and systems* **12**, 106–122 (2017).
80. Richter, O. *et al.* DYNAP-SE2: a scalable multi-core dynamic neuromorphic asynchronous spiking neural network processor. *Neuromorphic Computing and Engineering* **4**, 014003 (2024).
81. Neckar, A. *et al.* Braindrop: A mixed-signal neuromorphic architecture with a dynamical systems-based programming model. *Proceedings of the IEEE* **107**, 144–164 (2018).
82. Boybat, I. *et al.* Neuromorphic computing with multi-membristive synapses. *Nature communications* **9**, 2514 (2018).
83. Andrade-Talavera, Y., Fisahn, A. & Rodríguez-Moreno, A. Timing to be precise? An overview of spike timing-dependent plasticity, brain rhythmicity, and glial cells interplay within neuronal circuits. *Molecular Psychiatry* **28**, 2177–2188 (2023).
84. Dale, H. *Pharmacology and nerve-endings* 1935.
85. Gerstner, W., Kistler, W. M., Naud, R. & Paninski, L. *Neuronal dynamics: From single neurons to networks and models of cognition* (Cambridge University Press, 2014).

86. Bi, G.-q. & Poo, M.-m. Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. *Journal of neuroscience* **18**, 10464–10472 (1998).
87. Petreanu, L., Mao, T., Sternson, S. M. & Svoboda, K. The subcellular organization of neocortical excitatory connections. *Nature* **457**, 1142–1145 (2009).
88. Larkum, M. A cellular mechanism for cortical associations: an organizing principle for the cerebral cortex. *Trends in neurosciences* **36**, 141–151 (2013).
89. Makino, H. & Komiyama, T. Learning enhances the relative impact of top-down processing in the visual cortex. *Nature neuroscience* **18**, 1116–1122 (2015).
90. Gillon, C. J. *et al.* Learning from unexpected events in the neocortical microcircuit. *BioRxiv*, 2021–01 (2021).
91. Jordan, R. & Keller, G. B. Opposing influence of top-down and bottom-up input on excitatory layer 2/3 neurons in mouse primary visual cortex. *Neuron* **108**, 1194–1206 (2020).
92. Urbanczik, R. & Senn, W. Learning by the dendritic prediction of somatic spiking. *Neuron* **81**, 521–528 (2014).