Editors' Suggestion Featured in Physics

Smooth Exact Gradient Descent Learning in Spiking Neural Networks

Christian Klos[*] and Raoul-Martin Memmesheimer[†]
Neural Network Dynamics and Computation, Institute of Genetics, University of Bonn, 53115 Bonn, Germany

(Received 19 December 2023; accepted 22 November 2024; published 13 January 2025)

Gradient descent prevails in artificial neural network training, but seems inept for spiking neural networks as small parameter changes can cause sudden, disruptive appearances and disappearances of spikes. Here, we demonstrate exact gradient descent based on continuously changing spiking dynamics. These are generated by neuron models whose spikes vanish and appear at the end of a trial, where it cannot influence subsequent dynamics. This also enables gradient-based spike addition and removal. We illustrate our scheme with various tasks and setups, including recurrent and deep, initially silent networks.

DOI: 10.1103/PhysRevLett.134.027301

Introduction—Biological neurons communicate via short electrical impulses called spikes [1]. Besides their overall rate of occurrence, the precise timing of single spikes often carries salient information [2–5]. Taking into account spikes is therefore essential for the modeling and the subsequent understanding of biological neural networks [1,6]. To build appropriate spiking network models, powerful and well-interpretable learning algorithms are needed. They are further required for neuromorphic computing, an aspiring field that develops spiking artificial neural hardware to apply them in machine learning. It aims to exploit properties of spikes such as event-based, parallel operation (neurons only need to be updated when they send or receive spikes) and the temporal and spatial (i.e., in terms of interacting neurons) sparsity of communication to achieve tasks with unprecedented energy efficiency and speed [7–9].

The prevalent approach for learning in nonspiking neural network models is to perform gradient descent on a loss function [10,11]. Importantly, during such learning the representations change continuously and in a predictable manner as the networks are compositions of functions that are continuous in the network parameters. The transfer of gradient descent learning to spiking networks is, however, problematic due to the all-or-none character of spikes: the appearance or disappearance of spikes is not predictable from gradients computed for nearby parameter values. This is because the gradient only accounts for changes in spike timing of those spikes present when it is computed. Thus, a systematic addition or removal of spikes via exact gradient descent is seemingly not possible. This can, for example, lead to permanently silent, so-called dead neurons [12,13] and to diverging gradients [14]. Further, the network dynamics after a spike appearance or disappearance may change in a disruptive manner [15–18]. This can result in discontinuous changes of the representations, which are given by the spike times, and of the loss during learning.

Nevertheless, there are two popular approaches for learning in spiking neural networks based on gradient descent. The first approach, surrogate gradient descent, assumes binned time and replaces the binary activation function with a continuous-valued surrogate for the computation of the gradient [19]. It thus sacrifices the crucial advantage of event-based processing and learning only from spikes and necessitates the computation of state variables in each time step as well as their storage [20] (but see [21]). Furthermore, the computed surrogate gradient is only an approximation of the true gradient. The second approach, spike-based gradient descent, computes the exact gradient of the loss by considering the times of existing spikes as functions of the learnable parameters [12,22]. It allows for event-based processing but relies on *ad hoc* measures to deal with spike appearances and disappearances and gradient divergence, in particular to avoid dead neurons [23–27].

Here, we show that disruptive appearances and disappearances of spikes can be avoided. Consequently, all network spike times vary continuously and in some network models even smoothly, i.e., continuously differentiably, with the network parameters. This allows us to perform nondisruptive, exact gradient descent learning, including, as we show, the systematic addition or removal of spikes.

Neuron model—The most frequently employed neuron models when learning spiking networks are variants of the leaky integrate-and-fire (LIF) neuron [14,18–24,26,28,60,61]. LIF neurons, however, suffer from the aforementioned disruptive spike appearance and disappearance. For example, spikes can appear in the middle of a trial due to a continuous, arbitrarily small change of an input weight or time [Figs. 1(a) and 1(b)]. Here and in the following, a trial refers to an individual run of an experiment with finite duration.

[*] Contact author: cklos@uni-bonn.de

[†] Contact author: rm.memmesheimer@uni-bonn.de

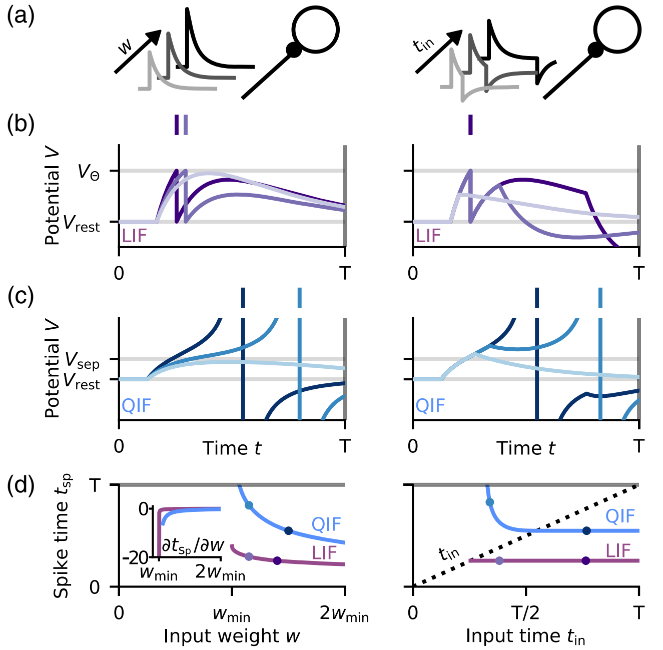

FIG. 1. Disruptive and nondisruptive appearance of spikes. (a),(b),(d) Spike times of the LIF neuron can appear disruptively in the middle of a trial. (a),(c),(d) Spike times of the QIF neuron only appear nondisruptively at the trial end and otherwise change continuously with changed parameters. Left column: a neuron receives a single input, whose weight is increased (traces with increasing saturation). Right column: a neuron receives an excitatory as well as an inhibitory input whose arrival is moved to larger times. (a) Setup (gray, different input currents). (b) LIF neuron membrane potentials [purple traces, saturation corresponding to (a); $V_{\rm rest}$ and V_Θ, resting and threshold potential; T, trial duration] and spikes (top, ticks). (c) Like (b) for the QIF neuron ($V_{\rm sep}$, separatrix potential). (d) Times of the first output spike as function of the changed parameter; dots correspond to equally colored spikes in (b),(c). Left: $w_{\rm min}$, weight at which the spike appears at finite time for the LIF neuron and at infinity for the QIF neuron. Inset: spike time gradient, divergent for the LIF neuron.

We therefore consider instead another important standard spiking neuron model, a quadratic integrate-and-fire (QIF) neuron [6,62,63]. Its membrane potential dynamics are governed by $\dot V = V(V-1)+I$, where I consists of temporally extended, exponentially decaying synaptic input currents, $\tau_{\rm s}\dot I = -I + \tau_{\rm s}\sum_i w_i \sum_{t_i}\delta(t-t_i)$. Here, $\tau_{\rm s}$ is the synaptic time constant, measured in multiples of the membrane time constant $\tau_{\rm m}=1$, and i indexes the presynaptic neurons, which spike at times t_i and have a synaptic weight w_i. In contrast to the LIF neuron, where $\dot V$ decays linearly with V, the QIF neuron explicitly incorporates the fact that in biological neurons the membrane potential further increases due to a self-amplification mechanism once it is large enough. As this generates spike upstrokes, the QIF neuron may be considered as the simplest truly spiking neuron model [63]. The voltage self-amplification is so strong that the voltage actually reaches infinity in finite time. One can define the time when this happens as the time of the spike, reset, and onset of synaptic transmission. We adopt this and henceforth call positive infinity the threshold of the QIF neuron for simplicity. For sufficiently negative voltage, the voltage increases strongly as well. The neuron can thus be reset to negative infinity, from where it quickly recovers. For LIF neurons, one needs to define finite threshold and reset potentials.

Nondisruptive appearance and disappearance of spikes and smooth spike timing—In the QIF neuron, spike times only appear and disappear at the trial end; otherwise they change smoothly with the network parameters. Importantly, this kind of spike appearance and disappearance is nondisruptive since the are no more spiking dynamics after the trial end that could be affected.

The mechanism underlying this feature can be intuitively understood: the voltage slope $\dot V$ at the threshold is infinitely large. If there is a small change, for example, in an input weight (Fig. 1, left column, blue curves), V and $\dot V$ will still be large close to where the spike has previously been. Therefore a spike will still be generated, only a bit earlier or later, unless it crosses the trial end. This is in contrast to the LIF neuron, where $\dot V$ at the threshold can tend to zero and a spike can therefore abruptly appear or disappear, accompanied by a diverging gradient (Fig. 1, left column, purple curves). A similar mechanism applies if there are changes in an input time as in Fig. 1, right column: an inhibitory input is moved backward in time until it crosses the time of an output spike generated by a sole, previous excitatory input [$t_{\rm in}$ crosses $t_{\rm sp}$ in Fig. 1(d) right]. In the QIF neuron, V and $\dot V$ are infinitely large at this point, such that the additional inhibitory input is negligible compared to the intrinsic drive. Thus there is no abrupt change in spike timing. In contrast, in the LIF neuron the inhibitory input induces a downward slope in the potential also if it is at the threshold. The spike induced by the excitatory input alone therefore suddenly appears once the inhibitory input arrives later.

In Supplemental Material, Sec. II [28], we prove the smoothness of the spike times and their nondisruptive appearance and disappearance in the general case with multiple inputs and output spikes.

Pseudodynamics and pseudospikes—The nondisruptive disappearance of spikes allows spike-based gradient descent to remove them in a controlled manner by shifting them past the trial end. In contrast, the gradient contains no information about spike appearances at the trial end, precluding the systematic addition of spikes. Being able to add spikes is, however, important because a neuron may initially or at some point during learning spike insufficiently often for the task or even be completely silent.

To solve this problem, we appropriately extend the ordinary dynamics by what we call pseudodynamics.

Concretely, we propose two types of pseudodynamics. In the first type, which we use in our applications, the neurons continue to evolve as QIF neurons, but with an added constant, suprathreshold drive, until they have spiked sufficiently often for the task [28]. We call the additional spikes pseudospikes. They only affect the pseudodynamics of postsynaptic neurons by controlling the value of the added drive. This ensures generically nonzero gradients. The continued evolution as a QIF neuron ensures continuity and mostly smoothness of the spike times, even if a spike transitions from a pseudospike to an ordinary one. In Supplemental Material, Sec. IB [28], we suggest a second approach where the spike times remain completely smooth.

Both types of pseudospike times have several useful properties: (i) they depend continuously and mostly smoothly on the network parameters, also when the pseudospikes cross the trial end to turn into ordinary spikes. (ii) If the voltage at the trial end increases, the pseudospike times decrease, intuitively because the neuron is already closer to spike. (iii) Pseudospikes affect postsynaptic pseudospikes but not ordinary ones. (iv) The pseudospikes interact such that the components of the gradient in multilayer networks are generically nonzero also if neurons are inactive during the actual trial duration. (v) The pseudospike times are computable in closed form.

Similar pseudospike time functions can be found for other neuron and synapse models with continuous spike times such as QIF neurons with infinitesimally short synaptic currents that generate voltage jumps [28].

Gradient descent learning—In the following, we apply spike-based gradient descent learning on the neural network models with continuous spike times identified above. We choose single neuron models with a closed-form solution between spikes and for the time of an upcoming spike. The former enables and the latter simplifies the use of efficient event-based simulations and modern automatic differentiation libraries [64,65] (The code to reproduce these results is publicly available [66].).

Interestingly, such solutions exist for the QIF neuron with temporally extended, exponentially decaying synaptic input currents if $\tau_s = \tau_m/2$ [28]. This is compatible with often assumed biologically plausible values, for example with a membrane time constant about 10 ms and a synaptic time constant about 5 ms [1,6]. In the examples in this Letter, we therefore use these values.

In the last of our three applications we employ oscillating QIF neurons with infinitesimally short input currents. Between spikes, they evolve with a constant rate of change in an appropriate phase representation [6,62,63,67], which further simplifies the event-based simulations. While their spike times are continuous, they are not smooth, as the derivative with respect to the time or weight of an input spike time jumps if it crosses another one.

Single neuron learning—As a first illustration of our scheme, we learn spike times of a single QIF neuron [Fig. 2(a); see [28] for details on models and tasks]. Initially it does not spike at all during the trial [Fig. 2(b), left]. We apply spike-based gradient descent to minimize the quadratic difference between two target and the first two output spike times (which may also be pseudospike times). The neuron is set to initially generate two pseudospikes, one for each target spike time. While not necessary in the displayed task, superfluous (pseudo)spikes can be included into the loss function with target behind the trial end to induce their removal if they enter the trial.

The use of pseudospikes allows one to activate the initially silent neuron [Fig. 2(c), gray background]. In doing so, the pseudospike times transition smoothly into ordinary spike times [Fig. 2(c), white background]. They are then shifted further until they lie precisely at the desired position on the time axis [Fig. 2(b), right]. The spike times change smoothly [Fig. 2(c)] and the loss gradient is continuous [Fig. 2(d)]. The example illustrates that our scheme allows one to learn precisely timed spikes of a single neuron in a smooth fashion and even if the neuron is initially silent.

Learning a recurrent neural network—Next, we consider the training of a recurrent neural network (RNN),

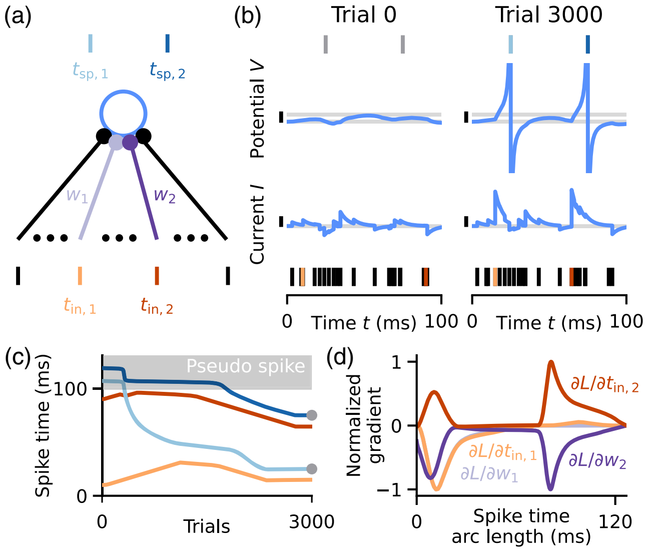

FIG. 2. Smooth gradient descent learning in a QIF neuron. (a) Weights (purple) and times (orange) of two inputs are learned to adjust the first two output spike times (blue). (b) Left: before learning, the neuron does not spike (gray ticks, target spike times; horizontal gray lines, V_{sep}, V_{rest}, or zero input current; black bars, potential or current difference of 1; orange, black ticks, learned, other input spikes). Right: after learning, the neuron spikes at the desired times (blue ticks cover gray ticks). (c) During learning, the [pseudo (gray area)] spike times change smoothly [colors as in (a); gray circles, target spike times]. (d) The components of the gradient of the loss function L change continuously during learning ($\partial L/\partial w_1$ mostly covered by $\partial L/\partial t_{in,1}$). Learning progress is displayed as a function of the arc length of the output spike time trajectories since the start of learning [28].

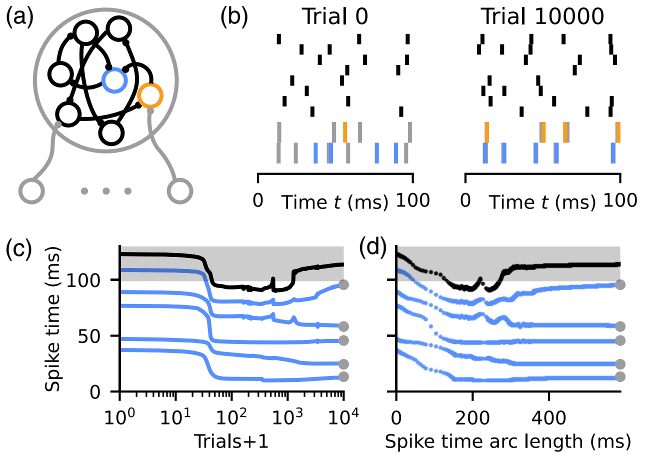

FIG. 3. Learning precise spikes in a RNN. (a) Network schematic. Neurons receive in each trial the same spikes from external input neurons (gray). Spike times of the first two network neurons are learned (blue and orange). (b) Spikes of network neurons before (left) and after (right) learning (colored ticks, spikes of first two neurons; gray ticks, target times). (c) Spike time trajectories of the first neuron during learning. Desired spikes (blue traces) shift toward their target times (gray circles). The first superfluous spike (black trace) is pushed out of the trial. (Gray area indicates pseudospikes.) (d) Same as (c) but the spike times are shown as a function of the arc length of the output spike time trajectories [28], which demonstrates their continuity, despite the occurrence of large gradients [cf. the steplike change in (c)].

where spike time changes have a global impact. It can be useful for the reconstruction of cortical networks [18,68,69]. We consider a fully connected RNN of ten QIF neurons with external inputs and learn the spike times of two network neurons by updating the recurrent weights and initial conditions [Fig. 3(a)]. In contrast to the learning of all network spikes [18,70], this does not reduce to independently finding a mapping from given input spike times to output spike times for each neuron.

Our scheme is successful also in this scenario and the spike times are precisely learned [Fig. 3(b)]. As in the previous example, the spike times of the first neuron change continuously during learning without discrete jumps [Figs. 3(c) and 3(d)]. Because of large gradients, which are typical for all kinds of RNNs [71], some changes are jumplike for the second neuron [28]. The underlying continuity becomes clear when restricting the maximal spike time change per step using adjustable update step sizes [28]. Hence, this example illustrates the applicability of our scheme to recurrent networks.

Standard machine learning task—Finally, we apply our scheme to the classification of hand-written single-digit numbers from the MNIST dataset, which is a widely used benchmark in neuromorphic computing (e.g., [20,24,61]).

We employ a three-layer feedforward network consisting of oscillatory QIF neurons with infinitesimally short input currents. For each input pixel, there is a corresponding input neuron, which spikes once at the beginning of the trial if the binarized pixel intensity is 1 and otherwise remains silent. The index of the neuron in the output layer that spikes first is the model prediction [72].

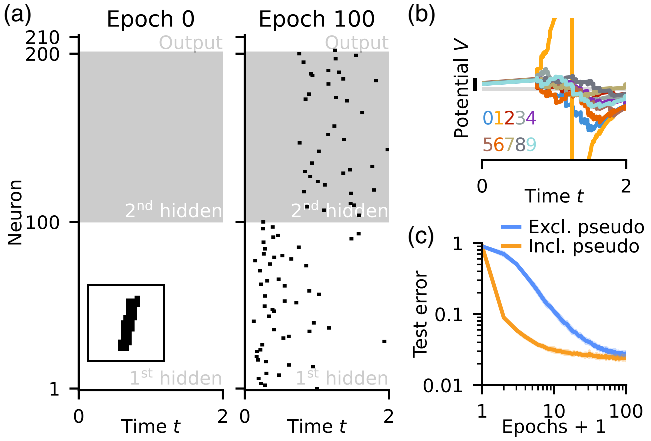

FIG. 4. MNIST task. (a) Spike raster plot of the three-layer network. Left: silent neurons before learning. Inset: example input also used on the right and in (b). Right: sparse spiking after learning. (b) Voltage dynamics of the output neurons after learning (horizontal gray line, V_{rest}; black bar, potential difference of 1). (c) Classification error dynamics. Utilizing pseudospikes also during testing (orange) generates smaller test errors in early training (solid lines indicate mean and shaded areas standard deviation over ten network instances).

To demonstrate that our scheme allows one to solve the dead neuron problem even if neurons in multiple layers are silent, we randomly initialize network parameters such that there are initially basically no ordinary spikes [Fig. 4(a), left]. Yet, the pseudospike time-dependent, imposed interaction between the neurons allows to backpropagate errors. Hence, the hidden and output neurons are activated [Figs. 4(a) right, (b)]. Finally basically all hidden neurons spike before the first output spike for some input image [28], indicating that they contribute to inference. Still activity is sparse. The final accuracy of 97.3% when only considering ordinary output spikes is comparable to previous results with similar setups [23–25,73]. If we also allow pseudospikes during testing (which only affects trials without ordinary output spikes), the accuracy does not change much. The minimal error level is, however, reached faster [Fig. 4(c)]. Thus, our scheme achieves competitive performance in a neuromorphic benchmark task even if almost no neurons are initially active.

Discussion—We have shown that there are neural networks with spike times that vary continuously or even smoothly with network parameters; ordinary spikes only appear and disappear at the trial end and can be extended to pseudospikes. The networks allow one to learn the timings of an arbitrary number of spikes in a continuous fashion with a spike-based gradient.

Perhaps surprisingly, the networks may consist of rather simple, standard QIF neurons. These are widely used in theoretical neuroscience [6,63], also for supervised learning [68,74,75], and have been implemented in neuromorphic hardware [76,77]. However, the particularity that spikes only appear and disappear at the trial end has not been noticed and exploited. We expect also that further neuron models exhibit spikes with continuous timings if their voltage slope close to the threshold is guaranteed to be positive. This includes neuron models that generate spikes by reaching infinite voltage, such as hybrid leaky integrate-and-fire neurons with an attached, nonlinear spike generation mechanism [78], the Izhikevich neuron with minor modifications [63], the rapid theta neuron [79,80], the sine neuron [81], and the exponential integrate-and-fire neuron [6]. It further includes intrinsically oscillating LIF neurons and antileaky integrate-and-fire neurons [82], if the impact of synaptic input currents vanishes at their spike threshold. We also expect that synapses with continuous current rise will be feasible, as well as conductance-based synapses.

On the one hand, our scheme possesses the same advantages as other spike-based gradient descent approaches such as small memory and computational footprints and a clear interpretation as following the exact loss gradient. On the other hand, like standard machine learning schemes it produces no disruptive transitions during learning and no gradient divergences. This suggests a wide range of applications: when studying biological neural networks, our scheme may be used to learn neurobiologically relevant tasks, to benchmark biological learning, to investigate how the network dynamical solutions may work, and to reconstruct synaptic connectivity from experimentally (partially) observed spiking activity. Furthermore, it may be used to train networks in neuromorphic computing (see [28] for further discussion). It generally allows one to benchmark other learning rules whose underlying mechanisms are less transparent and to train and pretrain networks before converting to a desired neuron type that complicates learning.

The dynamics of spiking and nonspiking neural networks can be chaotic [82–86] and give rise to exploding gradients [10,28,71,87]. We therefore restricted our learning examples to at most ten multiples of the membrane time constant. This fits the length of various experimentally observed precisely timed spike patterns [2,88–92] and the fast processing of certain tasks in neuromorphic computing [20,23–25,73].

Our pseudospikes allow the gradient to "see" spikes before they appear and to thus add spikes in a systematic manner. Pseudospikes affect the pseudospikes of postsynaptic neurons and ultimately of the output neurons. This preserves the gradients of the ordinary spike times and solves, in particular, the dead neuron problem. In a somewhat related approach, silent output neurons were assumed to spike at the trial end [26,27]. Our pseudospikes, however, apply to all neurons and allow one to backpropagate errors through silent neurons. The resulting possibility of initializing an entire network with small weights may be important to induce desirable and biologically plausible features such as energy-efficient final connectivity and sparse spiking [7,93], sparse coding [94], and representation learning [95].

To conclude, the present study shows something that seemed fundamentally impossible [8]: despite the inherent discreteness of spikes, there can be exact nondisruptive, even smooth gradient descent learning in spiking neural networks, including the gradient-based removal and, after augmentation with pseudodynamics, also generation of spikes.

Acknowledgments—We thank Sven Goedeke for helpful comments on the manuscript and the German Federal Ministry of Education and Research (BMBF) for support via the Bernstein Network (Bernsteinpreis 2014, 01GQ1710).

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

[2] T. Gollisch and M. Meister, Rapid neural coding in the retina with relative spike latencies, Science **319**, 1108 (2008).

[3] J. Wolfe, A. R. Houweling, and M. Brecht, Sparse and powerful cortical spikes, Curr. Opin. Neurobiol. **20**, 306 (2010).

[4] H. P. Saal, X. Wang, and S. J. Bensmaia, Importance of spike timing in touch: An analogy with hearing?, Curr. Opin. Neurobiol. **40**, 142 (2016).

[5] S. J. Sober, S. Sponberg, I. Nemenman, and L. H. Ting, Millisecond spike timing codes for motor control, Trends Neurosci. **41**, 644 (2018).

[6] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal Dynamics—From Single Neurons to Networks and Models of Cognition* (Cambridge University Press, Cambridge, England, 2014).

[7] M. Pfeiffer and T. Pfeil, Deep learning with spiking neurons: Opportunities and challenges., Front. Neurosci. **12**, 774 (2018).

[8] K. Roy, A. Jaiswal, and P. Panda, Towards spike-based machine intelligence with neuromorphic computing, Nature (London) **575**, 607 (2019).

[9] C. D. Schuman, S. R. Kulkarni, M. Parsa, J. P. Mitchell, P. Date, and B. Kay, Opportunities for neuromorphic computing algorithms and applications, Nat. Comput. Sci. **2**, 10 (2022).

[10] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, Cambridge, MA, 2016).

[11] N. Kriegeskorte and T. Golan, Neural network models and deep learning, Curr. Biol. **29**, R231 (2019).

[12] J. K. Eshraghian, M. Ward, E. Neftci, X. Wang, G. Lenz, G. Dwivedi, M. Bennamoun, D. S. Jeong, and W. D. Lu,

Training spiking neural networks using lessons from deep learning, arXiv:2109.12894.

[13] A. Taherkhani, A. Belatreche, Y. Li, G. Cosma, L. P. Maguire, and T. McGinnity, A review of learning in biologically plausible spiking neural networks, Neural Netw. **122**, 253 (2020).

[14] O. Booij and H. tat Nguyen, A gradient descent rule for spiking neurons emitting multiple spikes, Inf. Proc. Lett. **95**, 552 (2005).

[15] C. van Vreeswijk and H. Sompolinsky, Chaotic balanced state in a model of cortical circuits, Neural Comput. **10**, 1321 (1998).

[16] S. Jahnke, R.-M. Memmesheimer, and M. Timme, Stable irregular dynamics in complex neural networks, Phys. Rev. Lett. **100**, 048102 (2008).

[17] M. Monteforte and F. Wolf, Dynamic flux tubes form reservoirs of stability in neuronal circuits, Phys. Rev. X **2**, 041007 (2012).

[18] R.-M. Memmesheimer, R. Rubin, B. Ölveczky, and H. Sompolinsky, Learning precisely timed spikes, Neuron **82**, 925 (2014).

[19] E. O. Neftci, H. Mostafa, and F. Zenke, Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks, IEEE Signal Process. Mag. **36**, 51 (2019).

[20] T. C. Wunderlich and C. Pehle, Event-based backpropagation can compute exact gradients for spiking neural networks, Sci. Rep. **11**, 12829 (2021).

[21] N. Perez-Nieves and D. F. M. Goodman, Sparse spiking gradient descent, in *Advances in Neural Information Processing Systems*, edited by A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan (Curran Associates, Inc., New York, 2021).

[22] S. M. Bohte, J. N. Kok, and H. L. Poutré, Error-backpropagation in temporally encoded networks of spiking neurons, Neurocomputing **48**, 17 (2002).

[23] I. M. Comsa, K. Potempa, L. Versari, T. Fischbacher, A. Gesmundo, and J. Alakuijala, Temporal coding in spiking neural networks with alpha synaptic function, in *ICASSP 2020–2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE, Barcelona, 2020), pp. 8529–8533.

[24] J. Göltz, L. Kriener, A. Baumbach, S. Billaudelle, O. Breitwieser, B. Cramer, D. Dold, A. F. Kungl, W. Senn, J. Schemmel, K. Meier, and M. A. Petrovici, Fast and energy-efficient neuromorphic deep learning with first-spike times, Nat. Mach. Intell. **3**, 823 (2021).

[25] H. Mostafa, Supervised learning based on temporal coding in spiking neural networks, IEEE Trans. Neural Networks Learn. Syst. **29**, 3227 (2018).

[26] T. Nowotny, J. P. Turner, and J. C. Knight, Loss shaping enhances exact gradient learning with EventProp in spiking neural networks, arXiv:2212.01232.

[27] S. R. Kheradpisheh and T. Masquelier, Temporal backpropagation for spiking neural networks with one spike per neuron, International Journal of Neural Systems **30**, 2050027 (2020).

[28] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevLett.134.027301 for further model and task details, smoothness and continuity proofs, analysis of gradient statistics, further simulation results, and further discussion, which includes Refs. [29–59]

[29] T. P. Vogels, K. Rajan, and L. Abbott, Neural network dynamics, Annu. Rev. Neurosci. **28**, 357 (2005).

[30] A. Burkitt, A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input, Biol. Cybern. **95**, 1 (2006).

[31] W. R. Inc., Mathematica, Version 13.2, Champaign, IL, 2023.

[32] E. Kamke, *Differentialgleichungen. Lösungsmethoden und Lösungen* (Teubner, Stuttgart, 1977).

[33] H. Tuckwell, *Introduction to Theoretical Neurobiology: Volume 1. Linear Cable Theory and Dendritic Structure* (Cambridge University Press, Cambridge, England, 1988).

[34] H. Tuckwell, *Introduction to Theoretical Neurobiology: Volume 2. Nonlinear and Stochastic Theories* (Cambridge University Press, Cambridge, England, 1988).

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

[36] R.-M. Memmesheimer, Quantitative prediction of intermittent high-frequency oscillations in neural networks with supralinear dendritic interactions, Proc. Natl. Acad. Sci. U.S.A. **107**, 11092 (2010).

[37] R. Mirollo and S. Strogatz, Synchronization of pulse coupled biological oscillators, SIAM J. Appl. Math. **50**, 1645 (1990).

[38] R.-M. Memmesheimer and M. Timme, Designing complex networks, Physica D (Amsterdam) **224**, 182 (2006).

[39] M. D'Haene, B. Schrauwen, J. Van Campenhout, and D. Stroobandt, Accelerating event-driven simulation of spiking neurons with multiple synaptic time constants, Neural Comput. **21**, 1068 (2009).

[40] R. Brette *et al.*, Simulation of networks of spiking neurons: A review of tools and strategies, J. Comput. Neurosci. **23**, 349 (2007).

[41] C. R. Harris *et al.*, Array programming with NumPy, Nature (London) **585**, 357 (2020).

[42] P. Virtanen *et al.* (SciPy 1.0 Contributors), SciPy 1.0: Fundamental algorithms for scientific computing in Python, Nat. Methods **17**, 261 (2020).

[43] A. Paszke *et al.*, PyTorch: An imperative style, high-performance deep learning library, in *Advances in Neural Information Processing Systems*, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Curran Associates, Inc., New York, 2019), Vol. 32, pp. 8024–8035.

[44] I. Babuschkin *et al.*, *The DeepMind JAX Ecosystem* (2020), http://github.com/google-deepmind.

[45] P. Moritz, R. Nishihara, S. Wang, A. Tumanov, R. Liaw, E. Liang, M. Elibol, Z. Yang, W. Paul, M. I. Jordan, and I. Stoica, Ray: A distributed framework for emerging AI applications, in *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)* (USENIX Association, Carlsbad, CA, 2018), pp. 561–577.

[46] J. D. Hunter, Matplotlib: A 2D graphics environment, Comput. Sci. Eng. **9**, 90 (2007).

[47] M. A. Petroff, Accessible color sequences for data visualization, arXiv:2107.02270.

[48] W. Rudin, *Principles of Mathematical Analysis* (McGraw-Hill, New York, 1976).

[49] M. W. Hirsch and S. Smale, *Differential Equations, Dynamical Systems, and Linear Algebra*, Pure and Applied Mathematics Vol. 60 (Academic Press, San Diego [u.a.], 1974).

[50] G. Jetschke, *Mathematik der Selbstorganisation* (Harri Deutsch, Frankfurt am Main, 2009).

[51] V. I. Arnold, *Ordinary Differential Equations* (Springer, Berlin, 1992).

[52] H. Heuser, *Lehrbuch der Analysis. Teil 1* (Teubner-Verlag, Stuttgart, 1998).

[53] A. Stanojevic, S. Woźniak, G. Bellec, G. Cherubini, A. Pantazi, and W. Gerstner, An exact mapping from ReLU networks to spiking neural networks, Neural Netw. **168**, 74 (2023).

[54] A. Stanojevic, S. Woźniak, G. Bellec, G. Cherubini, A. Pantazi, and W. Gerstner, High-performance deep spiking neural networks with 0.3 spikes per neuron, arXiv: 2306.08744.

[55] T. Hoefler, D. Alistarh, T. Ben-Nun, N. Dryden, and A. Peste, Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks, J. Mach. Learn. Res. **22**, 10882 (2021).

[56] S. I. Mirzadeh, K. Alizadeh-Vahid, S. Mehta, C. C. del Mundo, O. Tuzel, G. Samei, M. Rastegari, and M. Farajtabar, ReLU strikes back: Exploiting activation sparsity in large language models, in *The 12th International Conference on Learning Representations* (OpenReview.net, Vienna, 2024).

[57] D. V. Christensen *et al.*, 2022 roadmap on neuromorphic computing and engineering, Neuromorphic Comput. Eng. **2**, 022501 (2022).

[58] E. Nordlie, M.-O. Gewaltig, and H. E. Plesser, Towards reproducible descriptions of neuronal network models, PLoS Comput. Biol. **5**, 1 (2009).

[59] J. Zhuang, T. Tang, Y. Ding, S. Tatikonda, N. Dvornek, X. Papademetris, and J. Duncan, AdaBelief optimizer: Adapting stepsizes by the belief in observed gradients, in *NeurIPS 2020 Workshop: Deep Learning through Information Geometry* (Curran Associates, Inc., New York, 2020).

[60] F. Zenke and S. Ganguli, SuperSpike: Supervised learning in multilayer spiking neural networks, Neural Comput. **30**, 1514 (2018).

[61] B. Cramer, S. Billaudelle, S. Kanya, A. Leibfried, A. Grübl, V. Karasenko, C. Pehle, K. Schreiber, Y. Stradmann, J. Weis, J. Schemmel, and F. Zenke, Surrogate gradients for analog neuromorphic computing, Proc. Natl. Acad. Sci. U.S.A. **119**, e2109194119 (2022).

[62] P. E. Latham, B. J. Richmond, P. G. Nelson, and S. Nirenberg, Intrinsic dynamics in neuronal networks. I. Theory, J. Neurophysiol. **83**, 808 (2000).

[63] E. Izhikevich, *Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting* (MIT Press, Cambridge, MA, 2007).

[64] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang, JAX: Composable transformations of Python + NumPy programs (2018), http://github.com/google/jax.

[65] R. Engelken, SparseProp: Efficient event-based simulation and training of sparse recurrent spiking neural networks, in *Advances in Neural Information Processing Systems*, edited by A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (Curran Associates, Inc., New York, 2023), pp. 3638–3657, Vol. 36.

[66] C. Klos (2024), https://github.com/chklos/spikegd.

[67] B. Ermentrout and N. Kopell, Parabolic bursting in an excitable system coupled with a slow oscillation, SIAM J. Appl. Math. **46**, 233 (1986).

[68] C. M. Kim and C. C. Chow, Learning recurrent dynamics in spiking networks, eLife **7**, e37124 (2018).

[69] A. Das and I. R. Fiete, Systematic errors in connectivity inferred from activity in strongly recurrent networks, Nat. Neurosci. **23**, 1286 (2020).

[70] R.-M. Memmesheimer and M. Timme, Designing the dynamics of spiking neural networks, Phys. Rev. Lett. **97**, 188101 (2006).

[71] R. Pascanu, T. Mikolov, and Y. Bengio, On the difficulty of training recurrent neural networks, in *Proceedings of the 30th International Conference on Machine Learning, Proceedings of Machine Learning Research* Vol. 28, edited by S. Dasgupta and D. McAllester (PMLR, Atlanta, GA, 2013), pp. 1310–1318.

[72] S. Thorpe, A. Delorme, and R. Van Rullen, Spike-based strategies for rapid processing, Neural Netw. **14**, 715 (2001).

[73] Y. Sakemi, K. Morino, T. Morie, and K. Aihara, A supervised learning algorithm for multilayer spiking neural networks based on temporal coding toward energy-efficient VLSI processor design, IEEE Trans. Neural Networks Learn. Syst. **34**, 394 (2023).

[74] D. Huh and T. J. Sejnowski, Gradient descent for spiking neural networks, in *Advances in Neural Information Processing Systems 31*, edited by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Curran Associates, Inc., New York, 2018), pp. 1439–1449.

[75] S. McKennoch, T. Voegtlin, and L. Bushnell, Spike-timing error backpropagation in theta neuron networks, Neural Comput. **21**, 9 (2009).

[76] E. Basham and D. Parent, An analog circuit implementation of a quadratic integrate and fire neuron, in *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society* (IEEE, New York, 2009).

[77] E. J. Basham and D. W. Parent, A neuromorphic quadratic, integrate, and fire silicon neuron with adaptive gain, in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* (IEEE, New York, 2018).

[78] M. Pospischil, Z. Piwkowska, T. Bal, and A. Destexhe, Comparison of different neuron models to conductance-based post-stimulus time histograms obtained in cortical pyramidal cells using dynamic-clamp *in vitro*, Biol. Cybern. **105**, 167 (2011).

[79] M. Monteforte, Chaotic dynamics in networks of spiking neurons in the balanced state, dissertation, University of Göttingen, 2011.

[80] R. Engelken, Chaotic neural circuit dynamics, dissertation, University of Göttingen, 2017.

[81] A. Viriyopase, R.-M. Memmesheimer, and S. Gielen, Analyzing the competition of gamma rhythms with delayed

pulse-coupled oscillators in phase representation, Phys. Rev. E **98**, 022217 (2018).

[82] P. Manz, S. Goedeke, and R.-M. Memmesheimer, Dynamics and computation in mixed networks containing neurons that accelerate towards spiking, Phys. Rev. E **100**, 042404 (2019).

[83] H. Sompolinsky, A. Crisanti, and H. J. Sommers, Chaos in random neural networks., Phys. Rev. Lett. **61**, 259 (1988).

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

[85] S. Jahnke, R.-M. Memmesheimer, and M. Timme, How chaotic is the balanced state?, Front. Comput. Neurosci. **3**, 13 (2009).

[86] M. Monteforte and F. Wolf, Dynamical entropy production in spiking neuron networks in the balanced state, Phys. Rev. Lett. **105**, 268104 (2010).

[87] R. Engelken, F. Wolf, and L. F. Abbott, Lyapunov spectra of chaotic recurrent neural networks, Phys. Rev. Res. **5**, 043044 (2023).

[88] Z. Nadásdy, H. Hirase, A. Czurkó, J. Csicsvari, and G. Buzsáki, Replay and time compression of recurring spike sequences in the hippocampus, J. Neurosci. **19**, 9497 (1999).

[89] R. S. Johansson and I. Birznieks, First spikes in ensembles of human tactile afferents code complex spatial fingertip events, Nat. Neurosci. **7**, 170 (2004).

[90] A. Luczak, P. Barthó, and K. D. Harris, Spontaneous events outline the realm of possible sensory responses in neocortical populations, Neuron **62**, 413 (2009).

[91] M. N. Havenith, S. Yu, J. Biederlack, N.-H. Chen, W. Singer, and D. Nikolic, Synchrony makes neurons fire in sequence, and stimulus properties determine who is ahead, J. Neurosci. **31**, 8570 (2011).

[92] A. Stella, P. Bouss, G. Palm, and S. Grün, Comparing surrogates to evaluate precisely timed higher-order spike correlations, eNeuro **9** (2022).

[93] C. Howarth, P. Gleeson, and D. Attwell, Updated energy budgets for neural computation in the neocortex and cerebellum, J. Cereb. Blood Flow Metab. **32**, 1222 (2012).

[94] B. Olshausen and D. Fields, Sparse coding of sensory inputs, Curr. Opin. Neurobiol. **14**, 481 (2004).

[95] T. Flesch, K. Juechems, T. Dumbalska, A. Saxe, and C. Summerfield, Orthogonal representations for robust context-dependent task performance in brains and neural networks, Neuron **110**, 1258 (2022).

Smooth Exact Gradient Descent Learning in Spiking Neural Networks — Supplemental Material —

Christian Klos* and Raoul-Martin Memmesheimer†

Neural Network Dynamics and Computation, Institute of Genetics, University of Bonn, 53115 Bonn, Germany

CONTENTS

I. MATERIALS AND METHODS

A. Neuron models

1. QIF neurons with extended coupling

We focus in our article on quadratic integrate-and-fire (QIF) neurons [1–3] that obey the ordinary differential equation

$$\dot{V}(t) = V(t)(V(t) - 1) + I(t). \tag{S1}$$

If V reaches infinity, $V(t_{\mathrm{sp}}^-) = V_\Theta = \infty$, an output spike is generated, and the voltage is reset to negative infinity, $V(t_{\mathrm{sp}}^+) = V_{\mathrm{reset}} = -\infty$. The superscripts $-$ and $+$ denote the limits from the left and right, respectively, which may be

* cklos@uni-bonn.de
† rm.memmesheimer@uni-bonn.de

interpreted as the times immediately before and after t_{sp}. For small V Eq. (S1) reduces to the LIF equation Eq. (S19) with dimensionless membrane time constant 1. Time is thus measured in multiples of the membrane time constant. Further, we have scaled and shifted the voltage such that without input the QIF neuron has a stable fixed point at the resting potential $V_{\mathrm{rest}} = 0$ and an unstable fixed point at the separatrix potential $V_{\mathrm{sep}} = 1$: For $I(t) = 0$ and $V_0 = V(0) = V_{\mathrm{rest}}$ or $V_0 = V_{\mathrm{sep}}$, the right hand side (rhs) of Eq. (S1) is zero, such that one has a fixed $V(t) = V_0$. If $V_0 < 0$, the rhs is positive and V increases towards V_{rest}. Similarly, if $V_{\mathrm{sep}} > V_0 > V_{\mathrm{rest}}$, the rhs is negative and V decreases towards V_{rest}. If $V_0 > V_{\mathrm{sep}}$ the rhs is positive, $V(t)$ accelerates towards infinity and a spike is generated. V_{sep} thus separates the two classes of trajectories with qualitatively different behavior.

For $I(t) = 0$ one can solve Eq. (S1) by separation of variables. With the initial condition $V(0) = V_0$ the time course of the voltage reads

$$V(t) = \frac{V_0}{V_0 - (V_0 - 1)\exp(t)}. \tag{S2}$$

Eq. (S2)'s rhs denominator, $V_0 - (V_0 - 1)\exp(t)$, is at $t = 0$ positive (equal to 1). If $V_0 > V_{\mathrm{sep}}$, it thereafter decreases as the subtrahend $(V_0 - 1)\exp(t)$ increases with time. The denominator becomes zero when t equals the spike time

$$t_{\mathrm{sp}} = \ln\left(\frac{V_0}{V_0 - 1}\right), \tag{S3}$$

such that $V(t_{\mathrm{sp}}) = \infty$. t_{sp} depends smoothly on V_0 and if V_0 tends to V_{sep}, t_{sp} tends to infinity.

The input current $I(t)$ consists of contributions due to spikes arriving from other neurons in the considered network. Additionally, there may be a constant input current component I_0, which covers average input from further neurons that are not explicitly modeled. To model temporally extended synaptic coupling, we implement standard current-based exponentially decaying synapses [4–6]. Specifically, at a spike arrival time t_i of a spike from neuron i, $I(t)$ increases about the strength w_i of the synapse from neuron i. Between spike arrivals, the current decays exponentially with time constant τ_{s}. $I(t)$ thus obeys

$$\tau_{\mathrm{s}}\dot{I}(t) = -(I(t) - I_0) + \tau_{\mathrm{s}}\sum_i w_i \sum_{t_i}\delta(t - t_i), \tag{S4}$$

with the Dirac delta distribution δ. We focus on neurons with $I_0 = 0$ in our article.

2. A closed-form solution

Interestingly, Eqs. (S1) and (S4) have a closed-form solution between spikes, if $\tau_{\mathrm{s}} = 1/2$ and $I_0 = 0$ (in general the solution involves Bessel functions [7]): To obtain it, we first shift the time origin to the beginning of the period of interest. Since the period of interest extends only to the next input or output spike, it does not contain spike arrivals and all synaptic input currents decay exponentially. Since, furthermore, all synaptic decay time constants are identical, this implies that we can gather the currents into a single exponentially decaying one: Eq. (S4) with $I_0 = 0$ yields

$$I(t) = \sum_i w_i \sum_{t_i \le 0} e^{-\frac{t-t_i}{\tau_s}} = \Big(\sum_i w_i \sum_{t_i \le 0} e^{-\frac{0-t_i}{\tau_s}}\Big) e^{-\frac{t-0}{\tau_s}} = I(0)e^{-\frac{t-0}{\tau_s}} = we^{-2t}, \tag{S5}$$

where we have called the current strength at the time origin $w = I(0)$ and inserted $\tau_s = 1/2$. Using this in Eq. (S1) gives

$$\dot{V}(t) = V(t)^2 - V(t) + we^{-2t}. \tag{S6}$$

The simple substitution $V(t) = e^{-t}u(t)$ leads to a differential equation for $u(t)$ where the variables separate,

$$\dot{u}(t) = (u^2(t) + w)e^{-t}, \tag{S7}$$

[8] (part C, Eq. (I·55)). The solution of Eq. (S6) with $V(0) = V_0$ is thus

$$V(t) = \begin{cases} \frac{V_0}{V_0 - (V_0 - 1)\exp(t)}, & \text{if } w = 0, \\ \sqrt{w}e^{-t}\tan\Big(\arctan\Big(\frac{V_0}{\sqrt{w}}\Big) + \sqrt{w}\,(1 - e^{-t})\Big), & \text{if } w > 0, \\ \operatorname{sgn}(V_0)\sqrt{-w}e^{-t}, & \text{if } w < 0 \text{ and } |V_0| = \sqrt{-w}, \\ \sqrt{-w}e^{-t}\coth\Big(\operatorname{arcoth}\Big(\frac{V_0}{\sqrt{-w}}\Big) - \sqrt{-w}\,(1 - e^{-t})\Big), & \text{if } w < 0 \text{ and } |V_0| > \sqrt{-w}, \\ \sqrt{-w}e^{-t}\tanh\Big(\operatorname{artanh}\Big(\frac{V_0}{\sqrt{-w}}\Big) - \sqrt{-w}\,(1 - e^{-t})\Big), & \text{if } w < 0 \text{ and } |V_0| < \sqrt{-w}. \end{cases} \tag{S8}$$

This solution yields closed-form conditions for the generation of output spikes and even closed-form expressions for the spike times. The case $w = 0$ is discussed in the previous paragraph. A spike is generated if $V_0 > V_{\mathrm{sep}} = 1$; Eq. (S3) provides the spike time. If $w > 0$, we have a spike under the condition that $\sqrt{w} + \arctan\left(\frac{V_0}{\sqrt{w}}\right) > \frac{\pi}{2}$: The argument of tan in the second line of Eq. (S8) is initially smaller than $\pi/2$ because $\arctan\left(\frac{V_0}{\sqrt{w}}\right) < \frac{\pi}{2}$ and the second summand is zero. The condition ensures that for time tending to infinity the argument exceeds $\pi/2$, since e^{-t} tends to zero. Therefore for some finite spike time t_{sp}, the argument reaches $\pi/2$ from below and tan and $V(t)$ tend to positive infinity when t_{sp} is approached. Setting the argument equal to $\pi/2$ yields

$$t_{\mathrm{sp}} = -\ln\left(1 - \frac{\pi}{2\sqrt{w}} + \frac{1}{\sqrt{w}}\arctan\left(\frac{V_0}{\sqrt{w}}\right)\right). \tag{S9}$$

For $w < 0$, there is no spike generation if $|V_0| \leq \sqrt{-w}$, because the solutions are bounded by $\sqrt{-w}$. If $V_0 > \sqrt{-w}$ there a spike is generated under the condition that $\operatorname{arcoth}\left(\frac{V_0}{\sqrt{-w}}\right) - \sqrt{-w} < 0$ holds: the argument of coth in the third line of Eq. (S8) is initially positive, since arcoth is positive for arguments larger than 1. The condition ensures that for time to infinity the argument becomes smaller than zero, since e^{-t} tends to zero. Therefore the argument reaches zero at a finite time from the positive side such that coth and $V(t)$ tend to positive infinity. This happens at

$$t_{\mathrm{sp}} = -\ln\left(1 - \frac{1}{\sqrt{-w}}\operatorname{arcoth}\left(\frac{V_0}{\sqrt{-w}}\right)\right). \tag{S10}$$

3. Phase representation

For the second type of pseudospike times (Sec. I B) and for our analytical considerations (Sec. II), we transform the voltage of QIF neurons with extended coupling to an angle variable. In other words, we transform the QIF neuron to a θ-neuron [1–3, 9]. The transformation is smooth, i.e. continuously differentiable, and bijective, except at spiketimes, where V becomes infinitely large and is reset. Concretely, we use

$$\phi = \Phi(V) = \frac{1}{\pi}\arctan\left(\frac{V}{\pi}\right) + \frac{1}{2}, \tag{S11}$$

such that the threshold and reset of ϕ are $\phi_\Theta = 1$ and $\phi_{\mathrm{reset}} = 0$. Identifying the phases of threshold and reset with each other lets the ϕ-dynamics take place on a circle, S^1. They obey the differential equation

$$\dot{\phi}(t) = \frac{1}{\pi}\frac{\dot{V}(t)/\pi}{1 + (V(t)/\pi)^2} = \cos(\pi\phi(t))\left(\cos(\pi\phi(t)) + \frac{1}{\pi}\sin(\pi\phi(t))\right) + \frac{1}{\pi^2}\sin^2(\pi\phi(t))I(t), \tag{S12}$$

where we used Eqs. (S11) and (S1) and $V = \Phi^{-1}(\phi) = -\pi\cot(\pi\phi)$. The point $\phi = 1$, which is the same as $\phi = 0$, is not particularly special anymore, as the right hand side of the differential equation is infinitely often continuously differentiable there. ϕ's temporal derivative at this point equals 1, independent of I.

4. QIF neurons with infinitesimally short coupling

Furthermore, we consider QIF neurons with input currents of infinitesimally short extent [5, 10–14]. These induce a jump-like response in the voltage upon input arrival. Specifically, at a spike arrival from neuron i, $V(t)$ increases by the synaptic strength w_i. $V(t)$ and $I(t)$ are thus determined by

$$\tau_{\mathrm{m}}\dot{V}(t) = V(t)(V(t) - 1) + I(t), \tag{S13}$$

$$I(t) = I_0 + \tau_{\mathrm{m}}\sum_i w_i \sum_{t_i}\delta(t - t_i). \tag{S14}$$

Here, τ_{m} is the membrane time constant, I_0 is the constant input current component and, as before, the voltage threshold is $V_\Theta = \infty$ and the reset potential $V_{\mathrm{reset}} = -\infty$.

In our simulations, we always use a suprathreshold constant input current, i.e. $I_0 > 1/4$, which ensures that $\dot{V}(t)$ is positive if there is no further input. Hence, the neurons are intrinsically oscillating. Their dynamics between spikes is simplified: they have no fixed points anymore and the voltage is always monotonously increasing. We transform the QIF neuron to a Θ-neuron, using the transformation

$$\phi = \Phi(V) = \frac{\tau_\mathrm{m}}{\sqrt{I_0 - \frac{1}{4}}}\left(\arctan\left(\frac{V - \frac{1}{2}}{\sqrt{I_0 - \frac{1}{4}}}\right) + \frac{\pi}{2}\right), \tag{S15}$$

$$V = \Phi^{-1}(\phi) = \sqrt{I_0 - \frac{1}{4}}\tan\left(\sqrt{I_0 - \frac{1}{4}}\frac{\phi}{\tau_\mathrm{m}} - \frac{\pi}{2}\right) + \frac{1}{2}. \tag{S16}$$

The threshold and reset of ϕ are then given by $\phi_\Theta = \Phi(\infty) = \tau_\mathrm{m}\pi/\sqrt{I_0 - \frac{1}{4}}$ and $\phi_\mathrm{reset} = \Phi(-\infty) = 0$, respectively. We choose a slightly different transformation than before (cf. Eq. (S11)), because it results in a constant phase velocity between spikes,

$$\dot{\phi}(t) = 1. \tag{S17}$$

The closed-form solution of Eq. (S17) between spikes and with $\phi(0) = \phi_0$ is simply $\phi(t) = \phi_0 + t$. If there are no spike arrivals, the next spike thus happens at $t_\mathrm{sp} = \phi_\Theta - \phi_0$. Such simple expressions are convenient for event-based simulations. At a spike arrival from neuron i at t_i, ϕ changes according to the transfer function or phase transition curve $H_w(\phi)$ [15–17]. Concretely,

$$\phi(t_i^+) = H_{w_i}(\phi(t_i^-)) = \Phi\left(\Phi^{-1}(\phi(t_i^-)) + w_i\right). \tag{S18}$$

5. LIF neurons with extended coupling

For comparison purposes, we also consider LIF neurons with extended coupling:

$$\dot{V}(t) = -V(t) + I(t), \tag{S19}$$

$$\tau_\mathrm{s}\dot{I}(t) = -(I(t) - I_0) + \tau_\mathrm{s}\sum_i w_i \sum_{t_i}\delta(t - t_i), \tag{S20}$$

where I_0 is the constant input current component and i indexes the presynaptic neurons with corresponding synaptic weights w_i and spike times t_i. Time, including the synaptic time constant τ_s, is measured in multiples of the membrane time constant τ_m and the voltage has been shifted and scaled such that the resting potential is at $V_\mathrm{rest} = 0$ and the threshold at $V_\Theta = 1$. Directly after reaching the threshold, the voltage is reset to $V_\mathrm{reset} = V_\mathrm{rest}$.

Assuming $I_0 = 0$ and $\tau_\mathrm{s} \neq \tau_\mathrm{m}$, the closed-form solution of Eq. (S19) with $V(0) = V_0$ and $I(t) = we^{-t/\tau_\mathrm{s}}$, i.e. between spikes, is given by

$$V(t) = V_0 e^{-t} + w\frac{\tau_\mathrm{s}}{1 - \tau_\mathrm{s}}(e^{-t} - e^{-\frac{t}{\tau_\mathrm{s}}}). \tag{S21}$$

If $\tau_\mathrm{s} = 1/2$, the rhs of Eq. (S21) is quadratic in e^{-t}, which allows to compute the time of the next spike, in case there is one, in closed form [18]. Specifically, the threshold crossing happens at

$$t_\mathrm{sp} = -\ln\left(\frac{1}{2w}\left(V_0 + w + \sqrt{(V_0 + w)^2 - 4wV_\Theta}\right)\right). \tag{S22}$$

Here we assumed that the argument of the logarithm lies between 0 and 1, which ensures that $V(t)$ reaches V_Θ.

B. Pseudospikes

1. First type of pseudospikes for QIF neurons with extended coupling

In this section, we explain the first type of pseudodynamics and pseudspikes for QIF neurons with extended coupling, cf. Sec. I A 1. For the pseudodynamics we assume that the neurons behave like freely evolving QIF neurons with an added, constant drive after the trial end. Specifically, we define them to be

$$\dot{V}_\mathrm{ps}(t) = V_\mathrm{ps}(t)(V_\mathrm{ps}(t) - 1) + \frac{1}{4} + g(I_\mathrm{ps}) \tag{S23}$$

with initial condition $V_{\mathrm{ps}}(T) = V(T)$, where T is the trial length. I_{ps} is a modified version of the input current at the trial end $I(T)$, see below, and $g(I) = \alpha \log(1 + \exp(I/\alpha))$ with a free parameter $\alpha > 0$. Choosing the pseudodynamics to also be quadratic ensures the smooth transition of ordinary spike times to pseudospike times (see Sec. III A 1). The added, suprathreshold drive $I_0 = \frac{1}{4} + g(I_{\mathrm{ps}})$ ensures that the pseudodynamics are oscillatory ($g(I_{\mathrm{ps}})$ is positive), such that pseudospikes are generated.

One can transform the voltage of the pseudodynamics with the same transformation as in Sec. I A 4 to an angle variable,

$$\phi_{\mathrm{ps}} = \Phi_{I_{\mathrm{ps}}}(V) = \frac{1}{\sqrt{g(I_{\mathrm{ps}})}} \left(\arctan\left(\frac{V - 1/2}{\sqrt{g(I_{\mathrm{ps}})}} \right) + \frac{\pi}{2} \right). \tag{S24}$$

The threshold and reset of ϕ_{ps} are then given by $\phi_{\Theta, I_{\mathrm{ps}}} = \pi/\sqrt{g(I_{\mathrm{ps}})}$ and $\phi_{\mathrm{reset}} = 0$, respectively. Eq. (S23) transforms to $\dot{\phi}_{\mathrm{ps}} = 1$, making the computability of the pseudospike time in closed form obvious. Specifically, the general expression for the time of the kth spike, in case it is a pseudospike, is

$$t_{\mathrm{ps}} = T + (k - n_{\mathrm{trial}})\phi_{\Theta, I_{\mathrm{ps}}} - \Phi_{I_{\mathrm{ps}}}(V(T)), \tag{S25}$$

where n_{trial} is the number of ordinary spikes. The factor $(k - n_{\mathrm{trial}})$ ensures continuity of spiketimes whenever the current or a previous spike time crosses the trial end (see Sec. III A 2 for details). For example, if an ordinary spike becomes a pseudospike, $-\Phi_{I_{\mathrm{ps}}}(V(T))$ jumps by $-\phi_{\Theta, I_{\mathrm{ps}}}$ since the reset crosses T. This is canceled by the simultaneous jump of $(k - n_{\mathrm{trial}})\phi_{\Theta, I_{\mathrm{ps}}}$ by $\phi_{\Theta, I_{\mathrm{ps}}}$, since n_{trial} decreases by one. The spiketimes t_{ps} thus change continuously.

To ensure generically non-zero gradients, the pseudospike times should be affected by other neurons even if they are not generating ordinary spikes. During the trial, a presynaptic spike leads to a jump of the input current about the synaptic weight. Inspired by this, we here assume that presynaptic neurons affect the constant input current I_0 by a fraction of the synaptic weight. Specifically, we set

$$I_{\mathrm{ps}} = I(T) + \sum_j w_j \frac{\Phi_{I_{\mathrm{ps},j}}(V_j(T))}{\phi_{\Theta, I_{\mathrm{ps},j}}}, \tag{S26}$$

where j indexes the presynaptic neurons. Thus, for each neuron j a fraction of its synaptic weight w_j is added to the input current at the trial end $I(T)$. This fraction depends on how close neuron j is to producing a spike at the trial end, reaching one when the neuron reaches the threshold there. The additional input ensures that errors can be backpropagated through silent neurons and guarantees continuity of I_{ps} in case a presynaptic spike from neuron j crosses the trial end: then $I(T)$ jumps by w_j, which is canceled because $V_j(T)$ jumps from ∞ to $-\infty$, which induces a jump in $\Phi_{I_{\mathrm{ps},j}}(V_j(T))$ by $-\phi_{\Theta, I_{\mathrm{ps},j}}$ (see Sec. III A 3 for details). Finally, one needs to specify the initial values of I_{ps} and the order in which Eq. (S26) is evaluated given the neural network architecture. Specifically, in a feedforward network, we set $I_{\mathrm{ps}} = I(T)$ for the neurons in the first layer and then use Eq. (S26) to sequentially compute I_{ps} for the other layers. In a recurrent network, we first set $I_{\mathrm{ps}} = 0$ for all neurons and then use Eq. (S26) to compute the final values of I_{ps}. These choices ensure the validity of the pseudospike properties listed in the main text.

The scaling factor in Eq. (S26) can be rewritten as

$$r_j = \frac{\Phi_{I_{\mathrm{ps},j}}(V_j(T))}{\phi_{\Theta, I_{\mathrm{ps},j}}} = \frac{t_{\mathrm{ps},j}^{\max} - t_{\mathrm{ps},j}}{t_{\mathrm{ps},j}^{\max} - T}. \tag{S27}$$

Here, $t_{\mathrm{ps},j}$ is the first pseudospike time of neuron j and

$$t_{\mathrm{ps},j}^{\max} = T + \phi_{\Theta, I_{\mathrm{ps},j}} \tag{S28}$$

is its latest possible timing, which occurs for $V_j(T) \to -\infty$. This shows that neurons with earlier first pseudospike have a stronger influence on the pseudospike times of their postsynaptic partners. Furthermore, Eqs. (S27), (S24) and (S26) show that r_i may be expressed as

$$r_i = f_i\big(\sum_j w_j r_j\big). \tag{S29}$$

Thus, we can compute the pseudospike times like the states in a network of rate neurons that is run for one time step. Comparing Eq. (S29) and Eq. (S27) yields the activation function

$$f_i(x) = \frac{\Phi_{I_{\mathrm{ps},i}}(V_i(T))}{\phi_{\Theta, I_{\mathrm{ps},i}}}\bigg|_{\sum_j w_j r_j = x} = \frac{1}{\pi} \arctan\left(\frac{V_i(T) - 1/2}{\sqrt{g(I_i(T) + x)}} \right) + \frac{1}{2}. \tag{S30}$$

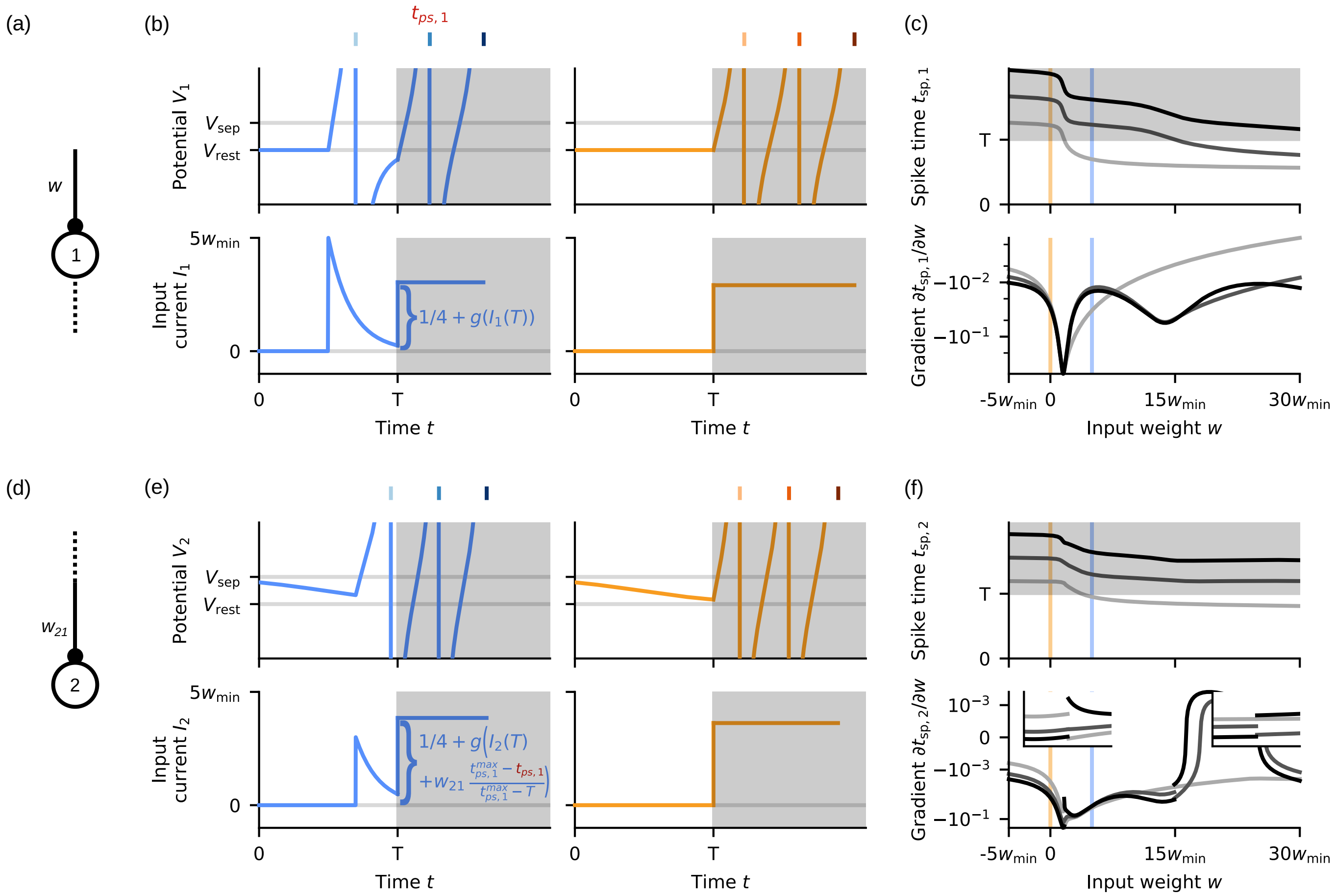

Figure S1. First type of pseudodynamics and pseudospikes. The figure shows in panels (a-c) results of simulations of a single neuron (neuron 1) that receives a single input spike; another neuron (neuron 2) is connected to neuron 1 and receives its output spikes (d-f). (a) Schematics of neuron 1, highlighting that it has a single input connection with weight w and a single output connection. (b) Ordinary and pseudodynamics of neuron 1 for two different weight values. Left, blue traces: $w = 5w_{\min}$, where $w_{\min}$ is the weight at which an ordinary spike appears at infinity (cf. main text Fig. 1). During the ordinary dynamics (white background), the input current (lower panel) due to the input spike is strong enough to induce an ordinary spike (upper panel, light blue vertical tick). Right, orange traces: $w = 0$, the neuron does not generate ordinary spikes. During the pseudodynamics (gray background) the input current is set to a constant, suprathreshold value. This value depends on the input current at the trial end, $I_1(T)$, and is therefore different for the blue and orange traces. The pseudodynamics start at $V_1(T)$ and generate pseudospikes. The pseudodynamics continue until the desired number of spikes is generated, which we here assume to be three. (c) Times of the three spikes of neuron 1 and their derivatives with respect to the input weight, as a function of the input weight. The spike times and their derivatives are continuous, which means that gradient descent can be used to smoothly shift spike times into the trial. Vertical lines correspond to similarly colored examples shown in (b). (d) Schematics of neuron 2, highlighting that it receives a connection with weight w_{21} from neuron 1. (e) Same as (b) but for neuron 2. The value of the input drive during the pseudodynamics depends on $I_2(T)$ and on the first pseudospike time of the presynaptic neuron 1 (dependency highlighted in red). Therefore it is different for the blue and orange traces. (f) Same as (c) but for neuron 2. The spike times are continuous and mostly smooth. Discontinuities of the derivatives of pseudospike times (insets) appear when a spike of the presynaptic neuron 1 crosses the trial end T. Gradient descent can be used to shift spike times into the trial even if neither neuron 2 nor neuron 1 spike during the trial.

In contrast to common networks of rate neurons, the activation function generally changes in each learning step, as it depends on $V_i(T)$ and $I_i(T)$.

In Sec. III A, we show the continuity and mostly smoothness of the here defined pseudospike times. Fig. S1 illustrates the computation of pseudospike times and the continuous and mostly smooth dependence of spike times on network parameters.

2. Second type of pseudospikes for QIF neurons with extended coupling

In the following, we explain the second type of pseudodynamics and pseudspikes. They are based on extending the dynamics behind the trial end, first according to the same differential equations that also govern the ordinary dynamics, thereafter according to simplified dynamics. For consistency with our explanation of the first type of pseudodynamics and pseudspikes, it makes most sense to call all dynamics and spikes within the trial duration $[0, T]$ ordinary dynamics and ordinary spikes. We use this convention in the main text; it implies that pseudospikes only influence pseudospikes, property (iii) in the main text. The pseudodynamics then consist of two parts, a first part obeying the same dynamical equations as the ordinary dynamics and a second part obeying modified dynamics. To simplify the subsequent explanations, we choose a different convention in the current section: We denote all dynamics that obey the ordinary dynamics equations by ordinary dynamics, even if they extend beyond T. Further, we call only the modified dynamics pseudodynamics.

The basic ideas behind the construction of the second type of pseudodynamics and pseudspikes may then be gathered as follows: (i) The ordinary neuronal dynamics guarantee smoothness of spikes, so we use it during the time when inputs arrive. (ii) (Active) Pseudospikes depend only on the phase at the end of the ordinary dynamics. (iii) If an ordinary spike disappears or appears a corresponding pseudospike appears or disappears at the beginning of the ensuing period of pseudodynamics. (iv) (Active) Pseudospikes that change to ordinary spikes are immediately replaced, such that there is always exactly one pseudospike per neuron. (v) The precise functional dependence of (active) pseudospike times on the phase at the end of the trial is such that spike times change smoothly with the network parameters also for special events like pseudospikes becoming ordinary ones.

In detail, we consider a feedforward network of L layers. The trial and thus the input spike trains last until T. The ordinary dynamics of the neurons in each layer beyond the first hidden layer are increasingly extended: in layer $l = 1, ..., L$ they last until

$$T_l = T + \frac{l-1}{d}, \tag{S31}$$

i.e. if we go up one layer, the ordinary dynamics last a fraction $1/d$ of the membrane time constant longer. We assume $d > 1$, which ensures the smoothness of spike times. After the ordinary dynamics, each neuron i in layer l generates pseudodynamics that lead to one pseudospike time

$$t_{\text{ps}} = T_l + \frac{1}{d} - \frac{1}{d}\phi(T_l)^d, \tag{S32}$$

where $\phi(T_l)$ is the phase Eq. (S11) at the end of the ordinary dynamics. If $\phi(T_l) = 1$, which is the same state as $\phi(T_l) = 0$, the value 0 is inserted into Eq. (S32), such that t_{ps} lies in the half open interval $(T_l, T_l + \frac{1}{d}]$ directly ensuing the period of ordinary dynamics. The pseudospike times from layer $l - 1$ thus arrive at the neurons of layer l towards the end of their ordinary dynamics. A pseudospike time t_{ps} in a neuron of layer l may be interpreted as resulting from completely externally driven pseudodynamics ϕ_{ps} beyond T_l. The continuous matching $\phi_{\text{ps}}(T_l^+) = \phi(T_l)$ to the preceding dynamics and the spike time condition $\phi_{\text{ps}}(t_{\text{ps}}) = 1$ imply that they can be specified as

$$\phi_{\text{ps}}(t) = \phi_{\text{ps}}(T_l^+) + 1 - (1 - d(t - T_l))^{\frac{1}{d}}, \tag{S33}$$

such that they obey the differential equation

$$\dot{\phi}_{\text{ps}}(t) = (1 - d(t - T_l))^{\frac{1}{d}-1}. \tag{S34}$$

Using Φ^{-1} (cf. Eq. (S11)), they can be transformed into voltage pseudodynamics, as displayed in Fig. S2a. Pseudodynamics with $d = 1$ linearly extrapolates the phase $\phi(T_l)$ to the threshold with slope one, such that the pseudospike happens at $t_{\text{ps}} = T_l + 1 - \phi(T_l)$.

If the network parameters change, pseudospikes become ordinary ones and vice versa. The related spiketimes change smoothly. For example, if a pseudospike of a neuron in layer l tends to T_l, $\phi(T_l)$ tends to 1, such that the ordinary spike appears at T_l exactly at the parameter value at which the pseudospike would reach T_l (and vanishes). The spiketime initially related to the pseudospike and then to the ordinary spike thus changes continuously. We assume that all pseudospikes that will be needed in the considered parameter range are held inactive but available at $T_l + \frac{1}{d}$. This may be important to construct a smooth cost function, because output layer spikes that are desired but not yet present as active pseudospikes can be included in it. (An alternative assumption compatible with our scheme is that a new pseudospike emerges if the current one becomes an ordinary spike.)

Fig. S2b,c illustrates the smooth dependence of the spiketimes on the network parameters in presence of pseudospikes. One can prove that it holds also at the transitions between inactive pseudospikes, active pseudospikes and ordinary spikes using methods similar to those of Sec. II.

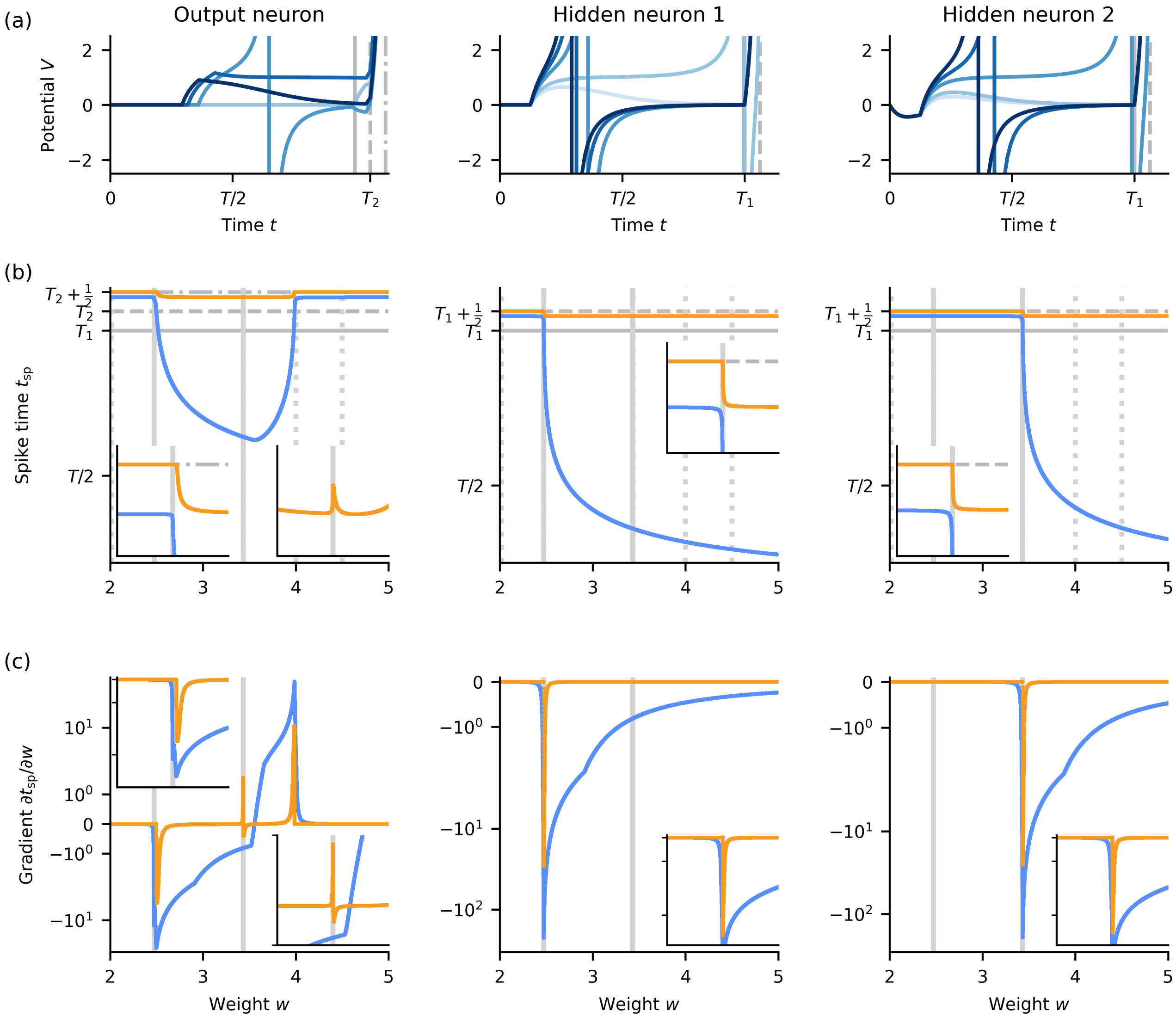

Figure S2. Second type of pseudodynamics and pseudospikes. The figure shows the results of simulations in a basic two-layer network with two hidden neurons and one output neuron. There is one input at the beginning of the trial, which inhibits hidden neuron 2, and one input a bit later, which excites both hidden neurons by w. Hidden neuron 1 excites the output neuron, hidden neuron 2 inhibits it. (a) Voltage traces of the output and the two hidden neurons for increasing w plotted in increasing color intensity. The pseudodynamics with $d = 2$ take place within $(T_1, T_1 + 1/d]$ and $(T_2, T_2 + 1/d]$ in the hidden and the output neurons, respectively. Solid, dashed and dashed-dotted vertical gray lines indicate T_1, $T_1 + 1/d = T_2$ and $T_2 + 1/d$, respectively. (b) Spike times as a function of w (blue, orange: first, second spike time of the different neurons). For increasing w there are transitions from an active pseudospike to an ordinary spike and simultaneously from an inactive to an active pseudospike, first in hidden neuron 1 then in 2. The insets show closeups of the curves around the corresponding weight values ($w \approx 2.47, 3.43$, solid gray vertical lines; spike time axis magnifications differ). The spiking of the hidden neurons and its temporal change trigger similar transitions in the output neuron. Dotted and solid vertical lines indicate weight values of traces displayed in (a). (c) like (b) for the gradient of the spike times with respect to w. The curves in (b,c) are continuous, because the spike times are smooth in w. This holds in particular at the transitions between inactive and active pseudospikes and between active pseudospikes and ordinary spikes.

3. Pseudospikes for QIF neurons with infinitesimally short coupling

For the pseudospikes of QIF neurons with infinitesimally short coupling (Sec. I A 4), we take a similar approach as for the first type of pseudospikes of QIF neurons with extended coupling (Sec. I B 1). This ensures that the pseudospike times are continuous. Specifically, we define the pseudodynamics to be

$$\tau_{\mathrm{m}}\dot{V}_{\mathrm{ps}}(t) = V_{\mathrm{ps}}(t)(V_{\mathrm{ps}}(t) - 1) + I_0, \tag{S35}$$

where I_0 has the same value as in Eq. (S14). In other words, the neurons continue to evolve as during the trial, but without interactions.

Similar to Sec. I B 1, we assume that neurons interact at the trial end with each other in the same way as during the trial but with scaled connection weights. Therefore, we set the initial condition for the pseudodynamics to

$$V_{\mathrm{ps}}(T) = V(T) + \sum_j w_j \frac{\Phi(V_{\mathrm{ps},j}(T))}{\phi_\Theta} \tag{S36}$$

where j indexes the presynaptic neurons and $\Phi(V)$ as well as ϕ_Θ are defined as in Sec. I A 4. Hence, the time of the kth spike, in case it is a pseudospike, is

$$t_{\mathrm{ps}} = T + (k - n_{\mathrm{trial}})\phi_\Theta - H_{\sum_j w_j \frac{\Phi(V_{\mathrm{ps},j}(T))}{\phi_\Theta}}(\Phi(V(T))), \tag{S37}$$

where n_{trial} is the number of ordinary spikes and $H_w(\phi)$ is defined as in Sec. I A 4. Similar to Eqs. (S26) and (S24), Eq. (S36) implies that we can obtain the pseudospike times from the states $V_i(T)$ at the trial end and variables r_i that are computed like the states of a network of rate neurons,

$$r_i = \frac{\Phi(V_{\mathrm{ps},i}(T))}{\phi_\Theta} = \frac{\Phi\left(V_i(T) + \sum_j w_j \frac{\Phi(V_{\mathrm{ps},j}(T))}{\phi_\Theta}\right)}{\phi_\Theta} \tag{S38}$$

$$= \frac{\Phi\left(V_i(T) + \sum_j w_j r_j\right)}{\phi_\Theta} = f_i\big(\sum_j w_j r_j\big). \tag{S39}$$

C. Simulation details

We mostly use exact, event-based simulations [19], where one iterates over spikes using the closed-form solutions for the evolution of the dynamical variables and upcoming spike times, see Secs. I A 2, I A 4 and I A 5. In each iteration, at first the neuron that spikes next as well as the time of the next spike is determined. Second, the state of all neurons is evolved until the next spike time. Third, the state of the neurons postsynaptic to the spiking neuron is updated based on the synaptic mechanism. Finally, the state of the spiking neuron is reset. For numerical reasons, some minor approximations are necessary if the absolute value of the membrane potential gets very large (see next paragraph). In Figs. S2, S3, S4, S5 and S6, we use time step based simulations, employing a standard ordinary differential equation solver between input and output spikes, with event detection to detect threshold crossings.

We simulate QIF neurons with extended coupling mostly in V-space. For the event-based simulations, we neglect the effect of an incoming spike on the next spike time of a neuron, if the spike time is less than ε away, where $\varepsilon = 10^{-6}$. Further, we do not update V if it is greater than $1/\varepsilon$ anymore and after spike generation at positive infinity, we reset V to $-1/\varepsilon$. For numerical purposes these values are sufficiently close to $\pm\infty$. In Fig. S6, we employ time-step-based voltage and current simulations with a threshold of 10^5 and a reset of -10^5. Figs. S2, S3, S4 and S5 use time-step-based phase and current simulations with threshold 1 and reset 0.

We simulate QIF neurons with infinitesimally short coupling in ϕ-space using event-based simulations. We neglect the effect of an incoming spike on ϕ and thus also the next spike time, if ϕ is very close to the threshold, $\phi > \Theta - \varepsilon$, or very close to the reset $\phi < \varepsilon$, where $\varepsilon = 10^{-6}$.

We simulate LIF neurons with extended coupling in V-space using event-based simulations. This is possible since we set the synaptic time constant to half of the membrane time constant. In this case, a closed-form solution of the threshold crossing time is available, see Sec. I A 5.

We use Python for all our simulations and analysis. For the event-based simulations and the automatic differentiation, we use JAX [20]. For the time step-based simulations, we use NumPy [21] and SciPy [22]. Further, we use PyTorch [23] for data loading, Optax [24] for optimization and Ray [25] for hyperparameter search. For plotting, we use Matplotlib [26] with colorblind-friendly colors [27]. All simulations were run on a local workstation with consumer-grade CPU (AMD Ryzen 1800X) and GPU (NVIDIA GeForce RTX 3090). Code to reproduce the main results is publicly available [28].

D. Spike time arc length

In some of our figures, we plot the evolution of spike times during learning as a function of the arc length of the spike time trajectories. At trial n, this is the cumulative, absolute change of all learned spike times until n:

$$L_t(n) = \begin{cases} 0, & \text{if } n = 0, \\ \sum_{l=1}^{n} \sum_{i=1}^{N_{\text{tar}}} \sum_{k_i=1}^{N_{\text{tar},i}} |t_{k_i}^{l} - t_{k_i}^{l-1}|, & \text{else,} \end{cases} \tag{S40}$$

where l indexes the trial, i indexes the N_{tar} neurons whose spike times are learned, k_i indexes the $N_{\text{tar},i}$ learned spike times of neuron i and and $t_{k_i}^{l}$ is the time of spike k_i at trial l. In Figs. S7 and S8, we additionally smooth the spike times with a rectangular kernel of length 11 trials before computing the spike time arc length to reduce the effect of oscillations on $L_t(n)$.

II. NON-DISRUPTIVE (DIS-)APPEARANCE OF SPIKES AND SMOOTH SPIKE TIMING IN QIF NEURONS WITH EXTENDED COUPLING

The following section shows that in QIF neurons with temporally extended coupling the output spike times depend smoothly on the input spike times and the input weights and that spikes can only (dis-)appear at the trial end. We note that for clarity of the arguments, in this section we allow this trial end to be at temporal infinity. We conduct the proof for arbitrary synaptic time constant; for simplicity, we assume that there is no constant input current. The proof uses well-known facts from analysis and the theory of differential equations. We sketch it in the next subsection, Sec. II A. Thereafter we detail it in five subsections that build on each other: Sec. II B shows smooth dependence of later states and spike times on the initial states. The initial state of the input current may be interpreted as the weight strength of a single input that arrives at the initialization time. Sec. II C generalizes this result by separating time into intervals in each of which one input arrives at the beginning. Sec. II D shows smooth dependence of later states and spike times on the spike arrival times, which form the endpoints of the intervals. The two remaining subsections, Sec. II E and Sec. II F, generalize the obtained results to neurons where the input spike times can change order with each other and with output spike times.

A. Proof overview

For the proof it is helpful to transform $V(t)$ smoothly and bijectively to a phase variable $\phi(t)$ on a circle, i.e. we transform the QIF neuron to a θ-neuron [1–3, 9]. The momentary impact of the input current on the phase is then phase-dependent. The point of spike generation is in the ϕ-dynamics not special anymore, except for the fact that the impact of the input current becomes zero there. This means that the threshold crossing itself happens purely due to the intrinsic neuron dynamics and always with the same finite rate of change $\dot{\phi}$.

We start by considering the case where there are no input spikes and the initial conditions are varied. This entails the case of having a single input spike with varying weight (main text Fig. 1 left column). Assuming the neuron does spike at least once, the implicit function theorem [29] (thm. 9.28) together with the finite rate of change of ϕ at threshold crossing then implies that also the output spike times vary smoothly. The important difference to the LIF neuron is here the always positive rate of change of ϕ at threshold crossing, which hinders the (dis-)appearance of spikes in the middle of a trial and that the gradient tends to infinity upon changing w.

Next, we consider the case of multiple input spikes with varying weights and times. If no spikes (two input or an input and an output spike), change order, the neuron's state prior to a given output spike but after the previous spike depends smoothly on the input parameters due to the smooth neuron dynamics between spikes. The considered output spike time then depends smoothly on this state because of the argument made above. More care has to be taken if two spikes change order. However, the dependence of output spike times turns out to be nevertheless smooth. For two interchanging input spikes this is ultimately because the order in which simultaneous inputs are processed does not matter (as they simply add to the current I). If an input and an output spike change order (main text Fig. 1 right column), it is because the impact of the input current on ϕ vanishes at the time of spike generation, as mentioned above. This is an important difference to the LIF neuron and hinders the (dis-)appearance of spikes in the middle of a trial.

B. Smooth dependence of the spike times on previous states

In this subsection we consider a scenario similar to main text Fig. 1 left column, i.e. a QIF neuron, Eq. (S1), with an exponentially decaying input current,

$$\tau_{\mathrm{s}}\dot{I}(t) = -I(t), \tag{S41}$$

for $t \geq 0$. The input may just have arrived at $t = 0$. The parameters are the initial states, $V(0) = V_0$ and $I(0) = w$, which shall be both finite. We show that the states and output spike times depend smoothly on the parameters and that the output spikes appear for increasing input strength w at infinite time or at the end of the trial, T, if it is earlier.

For this, we transform V to an angle variable ϕ using Eq. (S11). At the point of threshold crossing, $\phi = 1$, which is the same state as $\phi = 0$. ϕ's temporal derivative at this point equals 1, independent of I and thus w. We further restrict w to some compact interval $[w_{\min}, w_{\max}]$ with $w_{\min} \leq 0 \leq w_{\max}$. The dynamics of ϕ and I are for $t > 0$ given by the smooth system of differential equations Eqs. (S12) and (S41), which is defined on the compact set $S^1 \times [w_{\min}, w_{\max}]$. The dynamics do not leave this set. The solutions $\binom{\phi(t)}{I(t)}$ thus exist for all times and depend

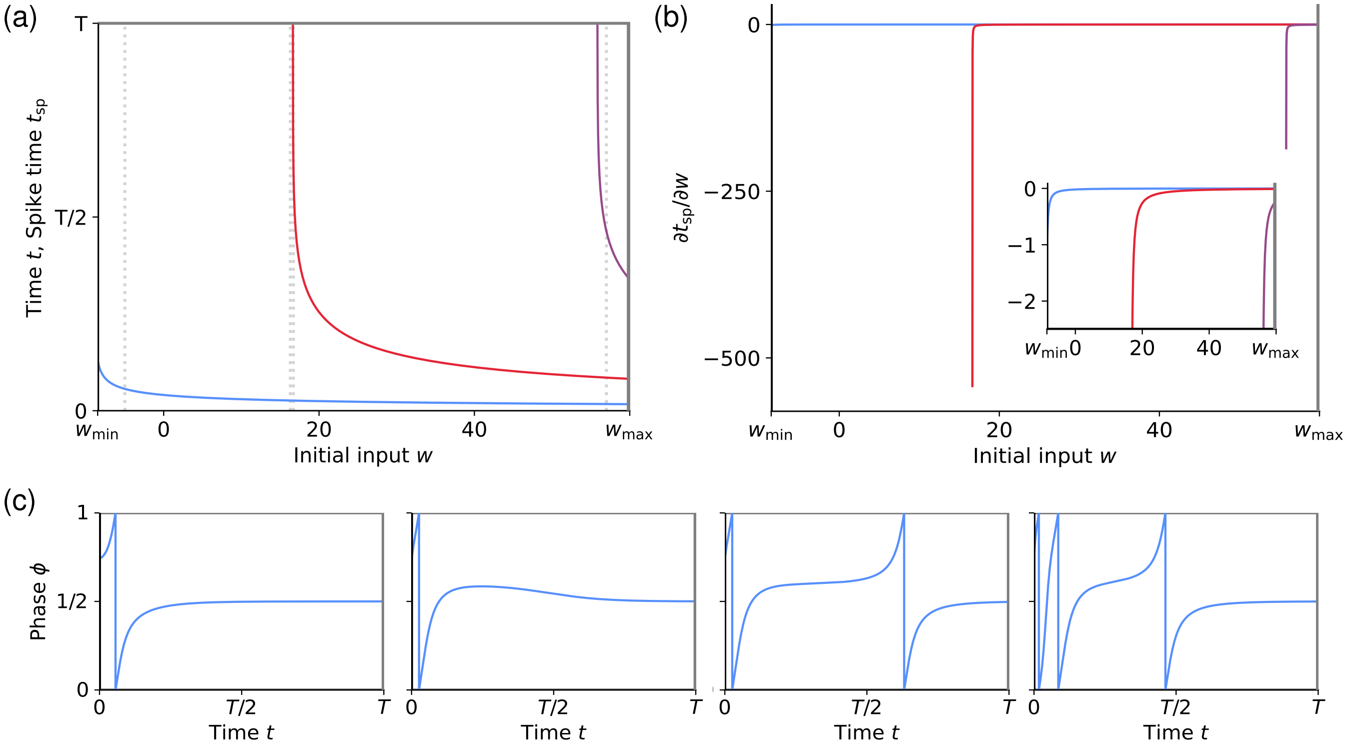

Figure S3. Spike times of a QIF neuron with a single exponentially decaying input arriving at $t = 0$. (a) The output spike times $t_{\rm sp}$ of the QIF neuron form continuous curves without kinks in (w, t)-space (blue, red, purple: first, second, third output spike time), which start at T or at $w_{\rm min}$ and end at $w_{\rm max}$ ($T = 10$, i.e. ten times the membrane time constant, $w_{\rm min} = -8.5, w_{\rm max} = 60$). They are the graphs of smooth functions $t_{\rm sp}(w)$. (b) Derivative of the output spike times with respect to w (blue, red, purple: derivative of first, second, third output spike time). $\frac{\partial t_{\rm sp}}{\partial w}$ is continuous. All derivative graphs start at finite values of $\partial t_{\rm sp}/\partial w$, since the trial duration T is finite. Starting points with $w > w_{\rm min}$ correspond to points where $t_{\rm sp}(w)$ starts to fall below T. Near these points, the derivatives assume large negative values. (c) Example traces $\phi(t)$ for different values of w (from left to right: $w = -5, 16.3, 16.7, 57$, highlighted by light gray vertical dotted lines in (a)) show first one and then a second and third spike. Spikes appear at the end of the trial and then shift to earlier times with increasing w.

smoothly on t and the initial conditions $\phi(0) = \phi_0 = \Phi(V_0)$ and $I(0) = w$, [30] (Secs. 8.5, 15.2), [31] (Sec. 1.2.3), [32] (Sec. 5.35).

Interpreting ϕ for fixed ϕ_0 as a function on (w, t)-space, we observe that the points (w, t) mapped to 1 specify the spike times $t = t_{\rm sp}$ of the neuron for the input strengths w, Fig. S3a. Since $\{1\}$ is a closed set and ϕ continuous, the preimage of $\{1\}$, i.e. the set of points (w, t) mapped by ϕ to 1, is closed as well. From the previous paragraph, we know that ϕ is even smooth in t and w and that the partial derivative with respect to t is at spike times invertible, since $\partial\phi/\partial t|_{(w,t_{\rm sp})} = 1 \neq 0$. The implicit function theorem [29] (thm. 9.28) thus ensures that the set of spike times in (w, t)-space looks locally, around each of its points, like the graph of a smooth function $t_{\rm sp}(w)$. The set thus consists of possibly multiple curves (for multiple spikes) in (w, t)-space, which are continuous, without "kinks" and with finite slope $dt_{\rm sp}/dw$, except where $t_{\rm sp}$ tends to infinity, Fig. S3a. The appearance of a spike corresponds to the start of such a curve. This start cannot lie in the interior of (w, t)-space, because the closeness implies that the starting point is part of the curve such that the implicit function theorem would guarantee continuation of the curve to both sides. The curves must thus extend to the borders of (w, t)-space. Specifically for growing w they start at $t = \infty$ or T or they start at $w = w_{\rm min}$, if ϕ_0 is so large that the spike is generated already for this input weight. They end at $w = w_{\rm max}$, because the spike times decrease monotonically with w, as $\dot{\phi}$ increases with increasing input. Spikes can for increasing w therefore only appear at $t = \infty$ or $t = T$. We note that the above argument also excludes merger of spike times, which would correspond to merger of curves. Further, an alike argument shows that $t_{\rm sp}$ depends smoothly on ϕ_0, w, Fig. S3b. The closed sets mapped by ϕ to 1 are then planes in ϕ_0, w, t-space. The above arguments do not apply to LIF neurons, since the temporal derivative of the voltage can become zero at spike times, see main text Fig. 1 left column.

C. Smooth dependence on input weights

The previous subsection showed that the membrane potential dynamics and the spike times of a QIF neuron with an exponentially decaying input depend smoothly on the initial conditions ϕ_0 and $I(0)$. We now turn to the case of multiple inputs and show smooth dependence of the output spike times t_{sp} on the synaptic input weights. If multiple spikes arrive, the input current Eq. (S41) changes in a jump-like manner by w_i at each arrival time t_i of a spike from neuron i,

$$\tau_{\mathrm{s}}\dot{I}(t) = -I(t) + \tau_{\mathrm{s}}\sum_i w_i \sum_{t_i} \delta(t-t_i). \tag{S42}$$

Note that for simplicity we use t_i for a single input spike, for all input spikes from neuron i and for input spikes in general. The jump-like change in I renders the value of I directly at t_i undefined, such that we need to separately consider the limits from below and above, $I(t_i^-)$ and $I(t_i^+)$. Further, it leads to finite size jumps in the temporal derivative of ϕ, but the value of ϕ itself still changes continuously. The previous subsection tells us that within the interval given by two subsequent spike times, t_i and t_j, the state and possible spike times depend smoothly on the state in its beginning, $\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^+)\end{smallmatrix}\right)$. This state results smoothly from the state at the end of the previous interval and the input weight, $\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^+)\end{smallmatrix}\right) = \left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^-)+w_i\end{smallmatrix}\right)$. $\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^-)\end{smallmatrix}\right)$, in turn, depends smoothly on the state at the beginning of the previous interval and so on. Thus, the state at any time t depends smoothly on the initial conditions at the very beginning and on the individual input weights. This implies that the partial derivatives of $\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^+)\end{smallmatrix}\right)$ with respect to each w_i are continuous. This holds irrespective of whether and when output spikes are generated, since the states where this happens, i.e. where $\phi(t_{\mathrm{sp}}) = 1$ holds, are not special for the neuron dynamics in ϕ, I-space. A function is continuously differentiable in all its variables exactly if all partial derivatives exist and are continuous [29] (thm. 9.21). This implies that because $\left(\begin{smallmatrix}\phi\\ I\end{smallmatrix}\right)$ is a smooth function of each single w_i, it is a smooth function of all w_i. For an output spike time $t_{\mathrm{sp}} \neq t_j$ for all j, Sec. II B shows that t_{sp} depends smoothly on closely nearby, previous states with no spike arrivals in between. The output spike time therefore also depends smoothly on all w_i. (If an input spike time agrees with an output spike time, $t_{\mathrm{sp}} = t_j$, the state is discontinuous in time as there is a jump in the current. We will see in Sec. II F that this does not cause problems, because the impact of inputs on ϕ vanishes at spike times.)

If a neuron receives at multiple times t_i input from the same input neuron i, the additive changes in I are the same, w_i, at these times. We have shown smooth dependence of $t_{\mathrm{sp}} \neq t_j$ for all j on the input weights of all input times, as if they were distinct variables. If some of these distinct variables have the same values and change in the same manner, t_{sp} still changes smoothly, which ensures smooth dependence of the output on the actual w_i.

D. Smooth dependence on input spike times

In our gradient descent scheme, also the input spike times to a neuron may change, for example because they are the output spike times of other neurons in the network. In the following we show that the output spike times of a neuron depend smoothly on the input spike times, if the order of (input and output) spike times stays the same. Since $\left(\begin{smallmatrix}\phi\\ I\end{smallmatrix}\right)$ depends smoothly on time between interval borders, $\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^-)\end{smallmatrix}\right)$ depends smoothly on t_i. The same holds for $\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^+)\end{smallmatrix}\right)$, since it differs from $\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^-)\end{smallmatrix}\right)$ only by a constant shift by w_i in I. Also the following states $\left(\begin{smallmatrix}\phi(t)\\ I(t)\end{smallmatrix}\right)$, $t > t_i$, and thus (cf. Sec. II B) the following output spike times $t_{\mathrm{sp}} \neq t_j$ then depend smoothly on t_i. For a preceding state (at a time $t < t_i$) and for preceding output spike times the smoothness property is trivially satisfied, since there is no dependence on t_i. (If t happens to agree with t_i, the state does not depend smoothly on t_i, because of the jump-like change in I.) Thus, as long as the output spike times satisfy $t_{\mathrm{sp}} \neq t_i$ and $t_{\mathrm{sp}} \neq t_j > t_i$, they depend smoothly on t_i, since there will always be states that depend smoothly on t_i so closely before t_{sp} that we can apply Sec. II C. (We note that since the times and states where an output spike is generated are not special for the neuron dynamics in ϕ, I-space, the agreement of other spike times t_j with other output spike times again does not change this.) Using also the results of the previous subsection, we conclude that as long as the spike order is conserved ($t_i \neq t_j$, $t_i \neq t_{\mathrm{sp}}$, $t_j \neq t_{\mathrm{sp}}$), states $\left(\begin{smallmatrix}\phi(t)\\ I(t)\end{smallmatrix}\right)$, $t \neq t_i$, and thus also the output spike times are a smooth function of each single w_i and t_i and thus of all of them.

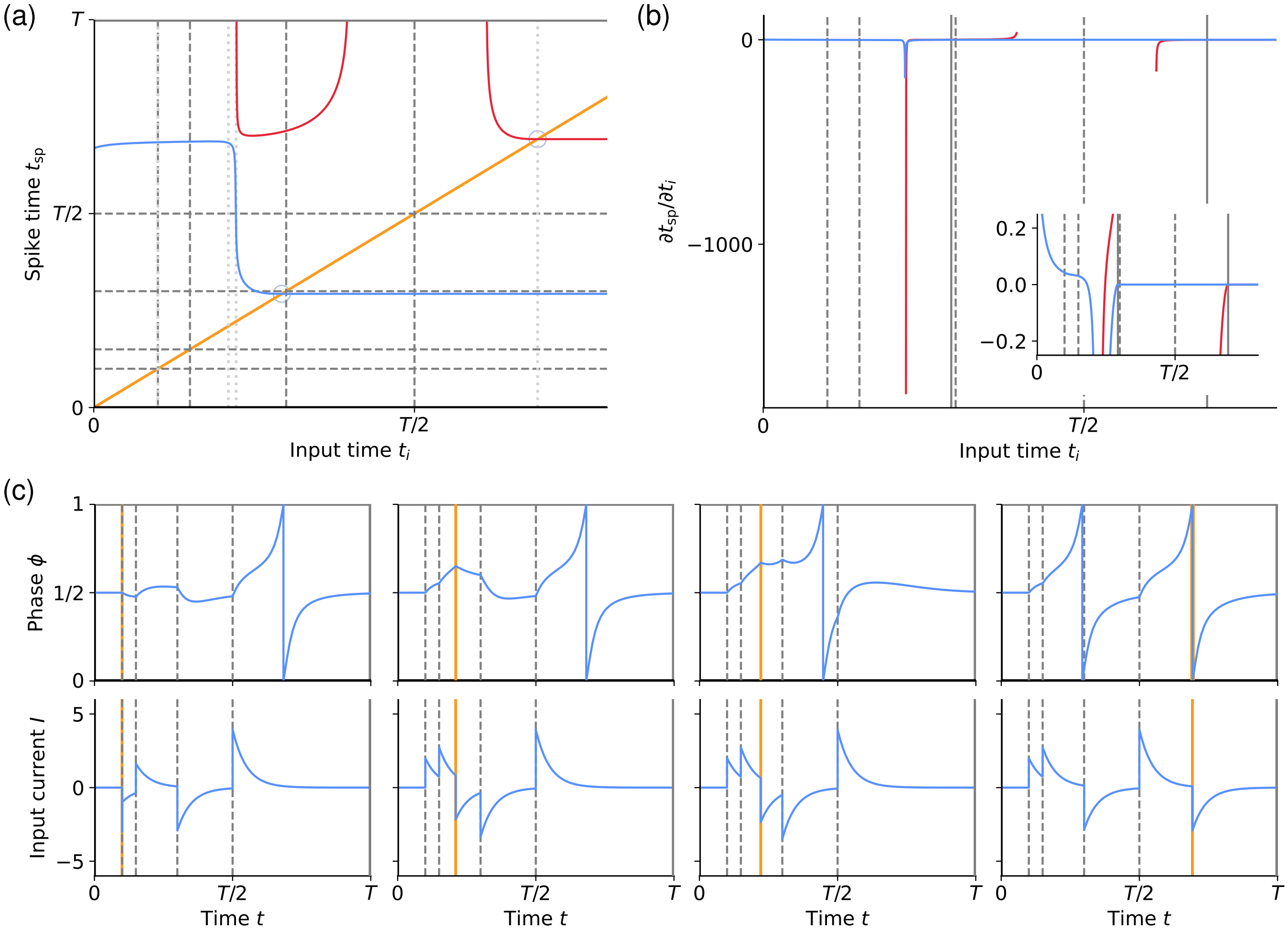

Figure S4. Change of output spike times when an input time changes. (a) The output spike times $t_{\rm sp}$ (blue, red: first, second output spike time) are smooth functions of the input spike time t_i. There are no jumps or kinks in the graphs, also if t_i crosses other input spike times t_j (gray dashed vertical lines: $t_i = t_j$) or if it crosses output spike times (orange diagonal: $t_{\rm sp} = t_i$, gray circles: crossing points of actual output spike times with t_i) or if the output spike times cross other input spike times t_j (gray dashed horizontal lines: $t_{\rm sp} = t_j$, partially crossed by blue curve). (b) The derivative $\partial t_{\rm sp}/\partial t_i$ confirms the smoothness of the function $t_{\rm sp}(t_i)$: It is continuous also at points where t_i crosses other t_j (gray dashed vertical lines) or where it agrees with actual output spike times (gray vertical lines). Inset: magnification of the range where derivatives are small, highlighting in particular the zero derivative when t_i is larger than $t_{\rm sp}$. The curves start and end at w values where $t_{\rm sp}(w)$ enters or exits the trial. (c) Example traces of $\phi(t)$ (upper panels) and $I(t)$ (lower panels) at different salient t_i values (highlighted by light gray dotted lines in (a); gray dashed vertical lines in (c): t_j, orange vertical line: t_i): at the crossing of t_i and a t_j (trace one: $t_i = 1$), closely before and after a fast change in the first spike time preceding an entering of the second spike time (traces two and three: $t_i = 2.1, 2.22$) and close to the crossing of the second output spike time and t_i (last trace: $t_i = 6.92$).

E. Changing input spike order

This subsection investigates whether we have smooth dependence of the output spike times on the input spike times when the order of the input spike times changes. Since the times of input and output spikes (henceforth, in short: events) form one-dimensional curves as a function of the training progress, interchanges of event order will generically happen, cf. Figs. S4 and S5. At a single point in the process, however, generically only two events cross. Therefore it suffices to only consider such cases here and in Sec. II F. Specifically the current subsection shows that the state at a test time t_2, which is so close after a pair of spikes t_i and t_j that there is no further input time between them, depends smoothly on t_i even if t_i just changes order with t_j, i.e. at $t_i = t_j$. Together with the results of Sec. II C (smooth dependence of subsequent on current states), Sec. II B (smooth dependence of output spike times on sufficiently closely preceding states), and Sec. II D (smooth dependence of the output spike times on t_i for $t_i \neq t_j$), this shows that the output spike times $t_{\rm sp}$ depend smoothly on a single t_i, if they do not change order with it, i.e. for all $t_i \neq t_{\rm sp}$. From

[29] (thm. 9.2), we again conclude that the output spike times depend smoothly on all t_i, as long as $t_i \neq t_{\mathrm{sp}}$.

For our considerations, it is convenient to introduce some further notions and abbreviations. First, we will use the flow [30–32] generated by the free differential equations Eqs. (S12) and (S41). This maps the state at t_a to the state at t_b, if there are no input spikes arriving in between. We denote the flow by $T_{t_b-t_a}\begin{pmatrix}\phi(t_a)\\ I(t_a^+)\end{pmatrix}$, such that

$$\begin{pmatrix}\phi(t_b)\\ I(t_b^-)\end{pmatrix} = T_{t_b-t_a}\begin{pmatrix}\phi(t_a)\\ I(t_a^+)\end{pmatrix}. \tag{S43}$$

We know from Sec. II B that this flow is a smooth, vector-valued function of its time and state argument. While ϕ is a continuous function of time, I is discontinuous at spike arrival times. Therefore, we regularly need to specify the left or right hand side time limits in the time argument of I as indicated: if t_a and t_b are subsequent spike arrival times, $T_{t_b-t_a}$ maps the state directly after t_a to the state directly before t_b. We further introduce the abbreviation f for the right hand side of the system of differential equations Eqs. (S12) and (S41) to compactly write

$$\begin{pmatrix}\dot\phi\\ \dot I\end{pmatrix} = f\begin{pmatrix}\phi\\ I\end{pmatrix}. \tag{S44}$$

As an immediate consequence of Eq. (S43) the time derivative of the flow is

$$\dot T_{t_b-t_a}\begin{pmatrix}\phi(t_a)\\ I(t_a^+)\end{pmatrix} = f\begin{pmatrix}\phi(t_b)\\ I(t_b^-)\end{pmatrix}. \tag{S45}$$

We will further use the derivative of the flow with respect to its state argument, the differential

$$DT_{t_b-t_a}\begin{pmatrix}\phi(t_a)\\ I(t_a^+)\end{pmatrix} = \begin{pmatrix}\frac{\partial\phi(t_b)}{\partial\phi(t_a)} & \frac{\partial\phi(t_b)}{\partial I(t_a^+)}\\ \frac{\partial I(t_b^-)}{\partial\phi(t_a)} & \frac{\partial I(t_b^-)}{\partial I(t_a^+)}\end{pmatrix}. \tag{S46}$$

Our aim is to show the smooth dependence of $\begin{pmatrix}\phi(t_2)\\ I(t_2)\end{pmatrix}$ on t_i at $t_i = t_j$. For this we first show the continuity of $\begin{pmatrix}\phi(t_2)\\ I(t_2)\end{pmatrix}$ and then the continuity of $\frac{\partial}{\partial t_i}\begin{pmatrix}\phi(t_2)\\ I(t_2)\end{pmatrix}$ as a function of t_i, at $t_i = t_j$. We start by considering the dynamics at $t_1 < \min(t_j, t_i)$, which shall be so close to t_i, t_j that there are no further input times between them, similar to $t_2 > \max(t_j, t_i)$. Three cases need to be distinguished: $t_i < t_j$, $t_i = t_j$ and $t_i > t_j$. In the first case, the state at t_2 may be written as

$$\left.\begin{pmatrix}\phi(t_2)\\ I(t_2)\end{pmatrix}\right|_{t_i<t_j} = T_{t_2-t_j}\left\{T_{t_j-t_i}\left(T_{t_i-t_1}\left[\begin{pmatrix}\phi(t_1)\\ I(t_1)\end{pmatrix}\right] + \begin{pmatrix}0\\ w_i\end{pmatrix}\right) + \begin{pmatrix}0\\ w_j\end{pmatrix}\right\}, \tag{S47}$$

in the second as

$$\left.\begin{pmatrix}\phi(t_2)\\ I(t_2)\end{pmatrix}\right|_{t_i=t_j} = T_{t_2-t_j}\left\{T_{t_j-t_1}\left[\begin{pmatrix}\phi(t_1)\\ I(t_1)\end{pmatrix}\right] + \begin{pmatrix}0\\ w_i+w_j\end{pmatrix}\right\}, \tag{S48}$$

and in the third as

$$\left.\begin{pmatrix}\phi(t_2)\\ I(t_2)\end{pmatrix}\right|_{t_i>t_j} = T_{t_2-t_i}\left\{T_{t_i-t_j}\left(T_{t_j-t_1}\left[\begin{pmatrix}\phi(t_1)\\ I(t_1)\end{pmatrix}\right] + \begin{pmatrix}0\\ w_j\end{pmatrix}\right) + \begin{pmatrix}0\\ w_i\end{pmatrix}\right\}. \tag{S49}$$

Here and in the following we employ also edged and curly brackets around function arguments to better distinguish them. To see the continuity of $\begin{pmatrix}\phi(t_2)\\ I(t_2)\end{pmatrix}$ as a function of t_i at $t_i = t_j$, we show that $\begin{pmatrix}\phi(t_2)\\ I(t_2)\end{pmatrix}$ converge to the same state, $\left.\begin{pmatrix}\phi(t_2)\\ I(t_2)\end{pmatrix}\right|_{t_i=t_j}$, when t_i approaches t_j from below or above,

$$\lim_{t_i\nearrow t_j}\left.\begin{pmatrix}\phi(t_2)\\ I(t_2)\end{pmatrix}\right|_{t_i<t_j} = T_{t_2-t_j}\left\{T_0\left(T_{t_j-t_1}\left[\begin{pmatrix}\phi(t_1)\\ I(t_1)\end{pmatrix}\right] + \begin{pmatrix}0\\ w_i\end{pmatrix}\right) + \begin{pmatrix}0\\ w_j\end{pmatrix}\right\} \tag{S50}$$

$$= T_{t_2-t_j}\left\{T_{t_j-t_1}\left[\begin{pmatrix}\phi(t_1)\\ I(t_1)\end{pmatrix}\right] + \begin{pmatrix}0\\ w_i+w_j\end{pmatrix}\right\} \tag{S51}$$

$$= \left.\begin{pmatrix}\phi(t_2)\\ I(t_2)\end{pmatrix}\right|_{t_i=t_j} \tag{S52}$$

$$= T_{t_2-t_j}\left\{T_0\left(T_{t_j-t_1}\left[\begin{pmatrix}\phi(t_1)\\ I(t_1)\end{pmatrix}\right] + \begin{pmatrix}0\\ w_j\end{pmatrix}\right) + \begin{pmatrix}0\\ w_i\end{pmatrix}\right\} \tag{S53}$$

$$= \lim_{t_i\searrow t_j}\left.\begin{pmatrix}\phi(t_2)\\ I(t_2)\end{pmatrix}\right|_{t_i>t_j}. \tag{S54}$$

The first, third and fifth line uses Eq. (S47), Eq. (S48) and Eq. (S49), respectively. The second and fourth lines use that the addition of weights commutes and that T_0 is the identity. The continuity at $t_i = t_j$ is thus a consequence of the fact that the addition of inputs to the current is commutative.

We will proceed similarly to see the continuity of the partial derivative $\frac{\partial}{\partial t_i}\left(\begin{smallmatrix}\phi(t_2)\\ I(t_2)\end{smallmatrix}\right)$. For this, we first compute the partial derivative for $t_i < t_j$ employing Eq. (S47), the chain rule and Eq. (S45),

$$\frac{\partial}{\partial t_i}\left(\begin{smallmatrix}\phi(t_2)\\ I(t_2)\end{smallmatrix}\right)\Big|_{t_i<t_j} = DT_{t_2-t_j}\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^+)\end{smallmatrix}\right)\cdot\Big\{-\dot{T}_{t_j-t_i}\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^+)\end{smallmatrix}\right)+DT_{t_j-t_i}\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^+)\end{smallmatrix}\right)\cdot\Big(\dot{T}_{t_i-t_1}\left(\begin{smallmatrix}\phi(t_1)\\ I(t_1)\end{smallmatrix}\right)\Big)\Big\} \tag{S55}$$

$$= DT_{t_2-t_j}\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^+)\end{smallmatrix}\right)\cdot\Big\{-f\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^-)\end{smallmatrix}\right)+DT_{t_j-t_i}\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^+)\end{smallmatrix}\right)\cdot f\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^-)\end{smallmatrix}\right)\Big\}. \tag{S56}$$

For $t_i > t_j$, we obtain from Eq. (S49)

$$\frac{\partial}{\partial t_i}\left(\begin{smallmatrix}\phi(t_2)\\ I(t_2)\end{smallmatrix}\right)\Big|_{t_i>t_j} = -\dot{T}_{t_2-t_i}\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^+)\end{smallmatrix}\right)+DT_{t_2-t_i}\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^+)\end{smallmatrix}\right)\dot{T}_{t_i-t_j}\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^+)\end{smallmatrix}\right) \tag{S57}$$

$$= -DT_{t_2-t_i}\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^+)\end{smallmatrix}\right)f\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^+)\end{smallmatrix}\right)+DT_{t_2-t_i}\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^+)\end{smallmatrix}\right)\dot{T}_{t_i-t_j}\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^+)\end{smallmatrix}\right) \tag{S58}$$

$$= DT_{t_2-t_i}\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^+)\end{smallmatrix}\right)\cdot\Big(-f\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^+)\end{smallmatrix}\right)+f\left(\begin{smallmatrix}\phi(t_i)\\ I(t_i^-)\end{smallmatrix}\right)\Big). \tag{S59}$$

The second line uses the general relation

$$\dot{T}_t(x) = \frac{d}{dr}T_{t+r}(x)\Big|_{r=0} = \frac{d}{dr}\,T_t\Big(T_r(x)\Big)\Big|_{r=0} \tag{S60}$$

$$= DT_t(x)\cdot\dot{T}_r(x)\Big|_{r=0} = DT_t(x)\cdot f(x). \tag{S61}$$

It reflects that we obtain the same state change if we (i) evolve the system about an infinitesimal interval dt past t (state change $\dot{T}_t(x)dt$) or if we (ii) evolve the initial state about dt (state change $f(x)dt$) and then evolve the change about t (via $DT_t(x)$, linear approximation suffices). We now compare the values of $\frac{\partial}{\partial t_i}\left(\begin{smallmatrix}\phi(t_2)\\ I(t_2)\end{smallmatrix}\right)$ when t_i approaches t_j from below or above. Eq. (S56) yields

$$\lim_{t_i\nearrow t_j}\frac{\partial}{\partial t_i}\left(\begin{smallmatrix}\phi(t_2)\\ I(t_2)\end{smallmatrix}\right)\Big|_{t_i<t_j} = DT_{t_2-t_j}\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^-)+w_i+w_j\end{smallmatrix}\right)\cdot\Big\{-f\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^-)+w_i\end{smallmatrix}\right)+DT_0\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^-)+w_i\end{smallmatrix}\right)\cdot f\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^-)\end{smallmatrix}\right)\Big\} \tag{S62}$$

$$= DT_{t_2-t_j}\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^-)+w_i+w_j\end{smallmatrix}\right)\cdot\Big\{-f\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^-)+w_i\end{smallmatrix}\right)+f\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^-)\end{smallmatrix}\right)\Big\}, \tag{S63}$$

where the limit in $I(t_j^-)$ is taken after the after the limit $t_i \nearrow t_j$, such that $I(t_j^-)$ is the current at t_j without both inputs w_i and w_j. From Eq. (S59) we obtain

$$\lim_{t_i\searrow t_j}\frac{\partial}{\partial t_i}\left(\begin{smallmatrix}\phi(t_2)\\ I(t_2)\end{smallmatrix}\right)\Big|_{t_i>t_j} = DT_{t_2-t_j}\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^-)+w_i+w_j\end{smallmatrix}\right)\cdot\Big(-f\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^-)+w_i+w_j\end{smallmatrix}\right)+f\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^-)+w_j\end{smallmatrix}\right)\Big). \tag{S64}$$

The right hand side of the system of differential equations Eqs. (S12) and (S41) is an affine map in I. It has the form

$$f\left(\begin{smallmatrix}\phi\\ I\end{smallmatrix}\right) = f_1(\phi)+f_2(\phi)I, \tag{S65}$$

with vector valued functions $f_1(\phi) = \left(\begin{smallmatrix}\cos(\pi\phi)\left(\cos(\pi\phi)+\frac{1}{\pi}\sin(\pi\phi)\right)\\ 0\end{smallmatrix}\right)$ and $f_2(\phi) = \left(\begin{smallmatrix}\frac{1}{\pi^2}\sin^2(\pi\phi)\\ -\frac{1}{\tau_\mathrm{s}}\end{smallmatrix}\right)$. The limits in Eq. (S63) and Eq. (S64) thus agree,

$$\lim_{t_i\nearrow t_j}\frac{\partial}{\partial t_i}\left(\begin{smallmatrix}\phi(t_2)\\ I(t_2)\end{smallmatrix}\right) = DT_{t_2-t_j}\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^-)+w_i+w_j\end{smallmatrix}\right)\cdot\Big\{-f_2(\phi(t_j))w_i\Big\} = \lim_{t_i\searrow t_j}\frac{\partial}{\partial t_i}\left(\begin{smallmatrix}\phi(t_2)\\ I(t_2)\end{smallmatrix}\right). \tag{S66}$$

Together with the continuity of $\left(\begin{smallmatrix}\phi(t_2)\\ I(t_2)\end{smallmatrix}\right)$ as a function of t_i around t_j (Eq. (S54), Sec. II D), this implies that the partial derivative $\frac{\partial}{\partial t_i}\left(\begin{smallmatrix}\phi(t_2)\\ I(t_2)\end{smallmatrix}\right)$ exactly at $t_i = t_j$ exists [33] (p. 286, Ex. 5) as well: it equals the limits Eq. (S66),

$$\frac{\partial}{\partial t_i}\left(\begin{smallmatrix}\phi(t_2)\\ I(t_2)\end{smallmatrix}\right)\Big|_{t_i=t_j} = DT_{t_2-t_j}\left(\begin{smallmatrix}\phi(t_j)\\ I(t_j^-)+w_i+w_j\end{smallmatrix}\right)\cdot\Big\{-f_2(\phi(t_j))w_i\Big\}. \tag{S67}$$

Intuitively the employed theorem indicates that a continuous function that is not differentiable has some kink; it can be proven using the mean value theorem [29] (thm. 5.10). We conclude that $\left(\begin{smallmatrix}\phi(t_2)\\ I(t_2)\end{smallmatrix}\right)$ depends smoothly on t_i also if it crosses other input spike times.

Since we can choose t_2 arbitrarily close to t_j, all states $\left(\begin{smallmatrix}\phi(t)\\ I(t)\end{smallmatrix}\right)$, $t \neq t_i$ depend smoothly on each and thus on all t_i, even if the input spikes change order with each other. As a consequence, the spike times $t_{\rm sp}$ are a smooth function of all $t_i \neq t_{\rm sp}$, even if these change order with each other, see Fig. S4. This is again because there are always states so closely before $t_{\rm sp}$ that there are no further input spikes in between, because these states depend smoothly on the t_i and because $t_{\rm sp}$ depends smoothly on them (Sec. II B).

F. Changing input and output spike order

In this subsection, we address input and output spike times that change order. This can happen because the input spikes change such that they cross output spikes and/or because the output spikes change (for example due to changes in previous input weights). Considering such crossings is particularly important, since also in a QIF neuron an inhibitory input usually leads to a downward jump in the voltage derivative and thus to a downward kink in the voltage (main text Fig. 1 right column). This kink, however, vanishes when t_i and $t_{\rm sp}$ cross, preventing disruptive spike (dis-)appearances like in the LIF neuron.

We consider an output spike time $t_{\rm sp}$ that tends to agree or agrees with an input spike time t_i. We first show that $t_{\rm sp}$ does not (dis-)appear in the middle of the trial and changes continuously and even smoothly as a function of previous $t_j < t_i$ and their weights w_j. For $t_j > t_i$ the property is obvious since there is no dependence on subsequent inputs, which is a special case of smooth dependence. Thereafter we show that $t_{\rm sp}$ does not (dis-)appear in the middle of the trial and changes continuously and even smoothly when t_i changes. $t_{\rm sp}$ cannot (dis-)appear and changes smoothly with the weight w_i associated with $t_i = t_{\rm sp}$, since changes in the input current I that take place at an output spike time leave the momentary phase $\phi(t_{\rm sp}) = 1$ and thus $t_{\rm sp}$ unaffected, Eq. (S12).

We first investigate whether $t_{\rm sp} = t_i$ may disruptively (dis-)appear and whether it changes continuously when previous $t_j < t_i$ change. For this we note that in contrast to Secs. II C and II D, we cannot simply use the implicit function theorem (via Sec. II B) to determine the properties of $t_{\rm sp}$, because at input arrivals $I(t)$ changes discontinuously, such that also $\phi(t)$ is not continuously differentiable with respect to time. This change, however, affects $\phi(t)$ only after t_i (for $t > t_i$). Therefore, if t_j tends to a limiting value $t_{j,0}$ such that $t_{\rm sp}$ tends to t_i from below, $t_{\rm sp}$ behaves like an output spike in a system without input at t_i. In particular, it depends smoothly on t_j and assumes the limiting value, $t_{\rm sp} = t_i$, if t_j assumes the limiting value, $t_j = t_{j,0}$ (Sec. II D). If t_j tends to $t_{j,0}$ such that $t_{\rm sp}$ tends to t_i from above, Sec. II B tells that $\phi(t_i) \nearrow 1$ and $\dot{\phi}(t_i^+) \to 1$, Eq. (S12). This implies that in the limit there is a threshold crossing at t_i, i.e. $t_{\rm sp} = t_i$ for $t_j = t_{j,0}$. Therefore, output spikes tending to t_i cannot vanish directly before reaching this limit, but continuously assume it. May an output spike vanish after reaching t_i, i.e. when $t_{\rm sp} = t_i$? To answer this we first note that the states $\left(\begin{smallmatrix}\phi(t)\\ I(t)\end{smallmatrix}\right)$ with t smaller or larger but sufficiently close (such that there are no further spike arrivals in between) to t_i, $t \lesssim t_i$ or $t \gtrsim t_i$, depend smoothly on t (Sec. II B). The same holds for the time derivative $\dot{\phi}(t)$, because it is a smooth function of $\phi(t)$ and $I(t)$, Eq. (S12). Further we know from Secs. II C and II D that the states $\left(\begin{smallmatrix}\phi(t)\\ I(t)\end{smallmatrix}\right)$ and thus $\dot{\phi}(t)$ with $t \approx t_i$ depend smoothly on previous t_j. We again denote by $t_{j,0}$ the value of t_j at which $t_{\rm sp} = t_i$. For $t_j = t_{j,0}$, $\phi(t_i) = 1$ and $\dot{\phi}(t_i^\pm) = 1$: the impact of the input at t_i vanishes and there is no kink in the phase despite the discontinuity of I at t_i. Due to the above mentioned smooth dependence on t we have $\dot{\phi}(t) \approx 1 > 0$ for $t \approx t_i$. Due to the smooth dependence of $\dot{\phi}(t)$ on t_j, it is positive also for $t_j \approx t_{j,0}$ and $\phi(t)$ is then a strictly monotonously increasing function of t for $t \approx t_i$. Therefore there is at most one threshold crossing. Furthermore, the values of $\phi(t)$ are close to their values for $t_j = t_{j,0}$ and $\phi(t)$ is continuous as a function of t. This guarantees a threshold crossing near t_i. We conclude that if there is a threshold crossing at t_i for $t_j = t_{j,0}$, also if t_j is unequal but sufficiently close to $t_{j,0}$ exactly one threshold crossing takes place, at a value $t_{\rm sp}$ near t_i. Because $t_j \to t_{j,0}$ implies $\phi(t_i) \to 1$ and $\dot{\phi}(t_i^\pm) \to 1$, we have $t_{\rm sp} \to t_i$, as already observed above. The spike time $t_{\rm sp}$ therefore does not disappear at t_i and changes continuously with t_j. We conclude that spikes cannot (dis-)appear at or in the direct vicinity of an input spike t_i due to continuous changes in previous input spike times. Furthermore output spike times $t_{\rm sp}$ depend continuously on previous input spike times t_j also if $t_{\rm sp}$ agrees with an input spike time, $t_{\rm sp} = t_i$. We can see analogously that the same holds for the weights w_j associated with t_j.

To show the existence and continuity of the derivative $\partial t_{\rm sp}/\partial t_j$ at $t_{j,0}$, we compute the derivatives $\partial t_{\rm sp}/\partial t_j$ for $t_{\rm sp}$ being close to but smaller or larger than t_i, $t_{\rm sp} \lesssim t_i$ or $t_{\rm sp} \gtrsim t_i$. We will observe that they tend to the same limit if t_j tends to $t_{j,0}$ such that $t_{\rm sp}$ tends to t_i from below or above. This implies existence and continuity of $\partial t_{\rm sp}/\partial t_j$ and the limit yields the value of this derivative at $t_{j,0}$ [33] (p. 286, Ex. 5), cf. also Sec. II E, Eq. (S67). If $t_{\rm sp} \lesssim t_i$ or $t_{\rm sp} \gtrsim t_i$,

$\partial t_{\rm sp}/\partial t_j$ can be computed using $\phi(t_{\rm sp}) - 1 = 0$ and the implicit function theorem,

$$\frac{\partial t_{\rm sp}}{\partial t_j} = -\frac{1}{\dot{\phi}(t_{\rm sp})}\frac{\partial \phi(t_{\rm sp})}{\partial t_j} = -\frac{\partial \phi(t_{\rm sp})}{\partial t_j}, \tag{S68}$$

where we have employed that always $\dot{\phi}(t_{\rm sp}) = 1$. If we choose again a reference time t_1 that is sufficiently close before t_i and $t_{\rm sp}$, we obtain for $t_{\rm sp} \lesssim t_i$,

$$\phi(t_{\rm sp}) = \left[T_{t_{\rm sp}-t_1}\begin{pmatrix}\phi(t_1)\\ I(t_1)\end{pmatrix}\right]_\phi, \tag{S69}$$

$$\frac{\partial \phi(t_{\rm sp})}{\partial t_j} = \left[DT_{t_{\rm sp}-t_1}\begin{pmatrix}\phi(t_1)\\ I(t_1)\end{pmatrix}\cdot\begin{pmatrix}\frac{\partial\phi(t_1)}{\partial t_j}\\ \frac{\partial I(t_1)}{\partial t_j}\end{pmatrix}\right]_\phi, \tag{S70}$$

$$\lim_{t_{\rm sp}\nearrow t_i}\frac{\partial \phi(t_{\rm sp})}{\partial t_j} = \left[DT_{t_i-t_1}\begin{pmatrix}\phi(t_1)\\ I(t_1)\end{pmatrix}\cdot\begin{pmatrix}\frac{\partial\phi(t_1)}{\partial t_j}\\ \frac{\partial I(t_1)}{\partial t_j}\end{pmatrix}\right]_\phi. \tag{S71}$$

$[.]_\phi$ means that we only take the first, ϕ-component of the final vector-valued expression. The limit in the last line occurs through t_j tending appropriately to $t_{j,0}$. If $t_{\rm sp} \gtrsim t_i$, we analogously have

$$\phi(t_{\rm sp}) = \left[T_{t_{\rm sp}-t_i}\left(T_{t_i-t_1}\begin{pmatrix}\phi(t_1)\\ I(t_1)\end{pmatrix} + \begin{pmatrix}0\\ w_i\end{pmatrix}\right)\right]_\phi, \tag{S72}$$

$$\frac{\partial \phi(t_{\rm sp})}{\partial t_j} = \left[DT_{t_{\rm sp}-t_i}\begin{pmatrix}\phi(t_i)\\ I(t_i^+)\end{pmatrix}\cdot DT_{t_i-t_1}\begin{pmatrix}\phi(t_1)\\ I(t_1)\end{pmatrix}\cdot\begin{pmatrix}\frac{\partial\phi(t_1)}{\partial t_j}\\ \frac{\partial I(t_1)}{\partial t_j}\end{pmatrix}\right]_\phi \tag{S73}$$

$$\lim_{t_{\rm sp}\searrow t_i}\frac{\partial \phi(t_{\rm sp})}{\partial t_j} = \left[DT_{t_i-t_1}\begin{pmatrix}\phi(t_1)\\ I(t_1)\end{pmatrix}\cdot\begin{pmatrix}\frac{\partial\phi(t_1)}{\partial t_j}\\ \frac{\partial I(t_1)}{\partial t_j}\end{pmatrix}\right]_\phi. \tag{S74}$$

In the last line we used that $DT_0\begin{pmatrix}\phi(t_i)\\ I(t_i^+)\end{pmatrix}$ is the identity matrix. The agreement of the partial derivatives' limits Eq. (S71) and Eq. (S74) reflects the fact that the states directly before and after t_i only differ by an addition of w_i, which moreover occurs to I, not to ϕ, such that the derivatives of $\phi(t_i^-)$ and $\phi(t_i^+)$ with respect to t_j are the same. The agreement shows the smooth dependence of $t_{\rm sp}$ on t_j at $t_j = t_{j,0}$, where $t_{\rm sp} = t_i$. An analogous consideration shows the existence and continuity of the derivative $\partial t_{\rm sp}/\partial w_j$ at $w_{j,0}$. We conclude that $t_{\rm sp}$ depends smoothly on earlier weights and spike times, also if it agrees with an input spike time.

We now study the only remaining case, the dependence of $t_{\rm sp}$ on t_i at $t_{\rm sp} = t_i$. We first assume that for t_i tending to $t_{i,0}$ from below, there is a spike $t_{\rm sp}$ tending to $t_{i,0}$. We ask if the spike will reach $t_{i,0}$ or whether it may disappear. This spike must occur after t_i, otherwise it cannot depend on t_i and converge to $t_{i,0} > t_i$. This implies that $\phi(t_i) \lesssim 1$, because the threshold crossing with $\phi(t_{\rm sp}) = 1$ is a bit later than t_i, and $\phi(t_i) \nearrow 1$. Again because the input at t_i affects the dynamics only after t_i, also in a modified system where this input is removed we have $\phi(t_i) \nearrow 1$ when $t_i \nearrow t_{i,0}$. In the modified system the phase dynamics are a smooth function of t around $t_{i,0}$. Thus, in the limit $t_i = t_{i,0}$ we have $\phi(t_i) = 1$ such that $t_{i,0}$ is a spike time of the modified system. If the input spike only arrives at $t_{i,0}$, the original and the modified systems' phases agree up to and including $t_{i,0}$. Therefore, also in the original system, we have $\phi(t_i) = 1$ for $t_i = t_{i,0}$. This implies that if $t_{\rm sp}$ tends to $t_{i,0}$ with $t_i \nearrow t_{i,0}$, $t_{\rm sp}$ also reaches the limit, $t_{\rm sp} = t_{i,0}$, for $t_i = t_{i,0}$. Now we consider the case that t_i tends to $t_{i,0}$ from above and $t_{\rm sp}$ tends to $t_{i,0}$. Since $t_i > t_{i,0}$ cannot influence $\phi(t_{i,0})$, we must have $\phi(t_{i,0}) = 1$, so $t_{\rm sp} = t_{i,0}$ for all the t_i tending to $t_{i,0}$. Since also an input at t_i does not change $\phi(t_i)$, for $t_i = t_{i,0}$ we have $\phi(t_{i,0}) = 1$ as well. $t_{\rm sp}$ therefore cannot suddenly disappear in the vicinity of $t_{i,0}$ due to t_i tending to and finally reaching $t_{i,0}$. Can a spike suddenly (dis-)appear at $t_{\rm sp} = t_{i,0}$ when $t_i = t_{i,0}$? If $t_{\rm sp} = t_i (= t_{i,0})$, the value of $\phi(t_{i,0})$ in the presence and in the absence of input at t_i are equal, again because the input w_i has no immediate impact on ϕ. (Moreover, the impact of any input vanishes at $\phi(t_{\rm sp}) = 1$.) Therefore $t_{i,0}$ is a spike time if the input at t_i is removed and for $t_i \geq t_{i,0}$. In the latter case, $t_{\rm sp}$ is constant as a function of t_i, in particular it does not vanish and depends smoothly on t_i. We thus consider $t_i \lesssim t_{i,0}$ in the following. In the absence of an input at t_i, the states sufficiently closely before $t_{i,0}$ are a smooth function of t. Further, since there is a threshold crossing at $t_{i,0}$, which implies a phase slope of $\dot{\phi}(t_{i,0}) = 1$, we have a phase that is slightly smaller than the threshold, $\phi(t) \lesssim 1$, for $t \lesssim t_{i,0}$. As a consequence, also in the system with input at t_i we have for $t_i \nearrow t_{i,0}$ that $\phi(t_i) \nearrow 1$ and $\dot{\phi}(t_i^-) \to 1$, Eq. (S12). Further, because the impact of an input goes to zero when approaching the threshold, we have $\dot{\phi}(t_i^+) \to 1$. The smoothness of the ϕ, I-dynamics behind t_i and the convergence to a nonzero $\dot{\phi}(t_i^+)$ implies that the ϕ-dynamics will reach 1 if the initial condition $\phi(t_i)$ is close enough to 1. This shows that for t_i sufficiently close to $t_{i,0}$ there will be a spike time $t_{\rm sp} \approx t_{i,0}$. Therefore spikes cannot appear at $t_i = t_{i,0}$. Furthermore, the threshold

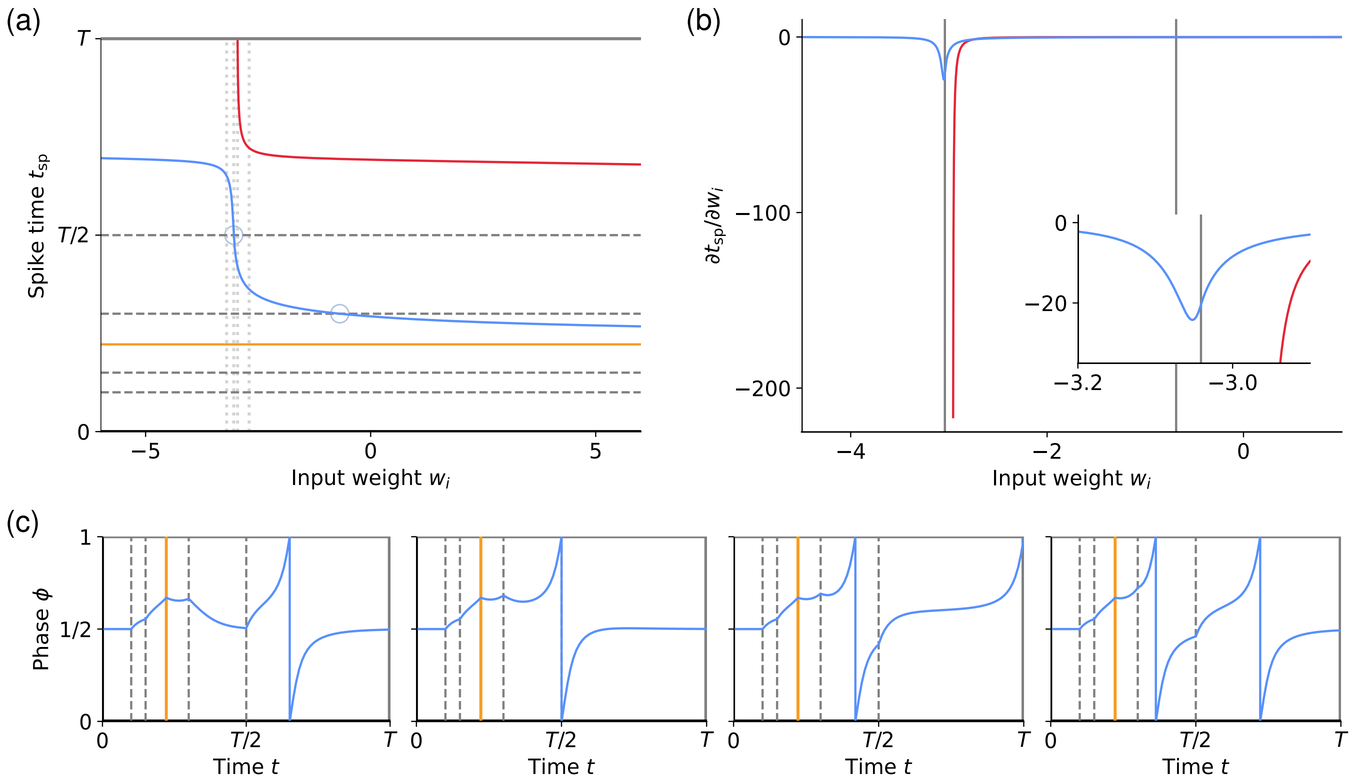

Figure S5. Change of output spike times when the strength of one of multiple inputs changes. (a) The output spike times $t_{\rm sp}$ (blue, red: first, second output spike time) are smooth functions of the input strength w_i arriving at t_i (orange horizontal line: $t_{\rm sp} = t_i$). There are no jumps or kinks in the graphs, also when $t_{\rm sp}$ crosses input spike times (gray dashed horizontal lines: $t_{\rm sp} = t_j$, partially crossed by blue curve). (b) The derivative $\partial t_{\rm sp}/\partial w_i$ confirms this smoothness. It is continuous also at values of w_i where the output spike times cross input spike times (gray vertical lines; inset: magnification of the region around the crossing with smallest w_i). (c) Example traces of $\phi(t)$ at w_i values around the fast change and the first crossing of the first $t_{\rm sp}$ with a t_j ($w_i = -3.2, -3.041, -2.957, -2.7$, highlighted by light gray dotted lines in (a); gray dashed vertical lines: t_j, orange vertical line: t_i).

crossing will be arbitrarily closely after t_i for $\phi(t_i)$ tending to 1. Therefore, the spike time $t_{\rm sp}$ converges to t_i and thus to $t_{i,0}$. We conclude that spikes cannot (dis-)appear at $t_{\rm sp} = t_i = t_{i,0}$ and $t_{\rm sp}$ is a continuous function of t_i at $t_{\rm sp} = t_i$.

Also the partial derivative $\frac{\partial t_{\rm sp}}{\partial t_i}$ is continuous at $t_i = t_{i,0}$, where $t_{\rm sp} = t_i$: We need to show that $\lim_{t_i \nearrow t_{i,0}} \frac{\partial t_{\rm sp}}{\partial t_i} = 0$, since $\lim_{t_i \searrow t_{i,0}} \frac{\partial t_{\rm sp}}{\partial t_i} = 0$ due to $t_{\rm sp}$'s independence of t_i for $t_i \geq t_{i,0}$ (where we have $t_{\rm sp} = t_{i,0}$, see the previous paragraph). We again choose a reference time t_1 so close before $t_i \lesssim t_{i,0}$ that there are no further inputs in between and t_i so close to $t_{i,0}$ that there is no spike arrival between t_i and the spike time $t_{\rm sp} \approx t_{i,0}$. Based on $\phi(t_{\rm sp}) - 1 = 0$ the implicit function theorem yields the derivative

$$\frac{\partial t_{\rm sp}}{\partial t_i} = -\frac{1}{\dot\phi(t_{\rm sp})}\frac{\partial \phi(t_{\rm sp})}{\partial t_i} = -\frac{\partial \phi(t_{\rm sp})}{\partial t_i} \tag{S75}$$

$$= -\frac{\partial}{\partial t_i}\left[T_{t_{\rm sp}-t_i}\left(T_{t_i-t_1}\left[\begin{pmatrix}\phi(t_1)\\ I(t_1)\end{pmatrix}\right] + \begin{pmatrix}0\\ w_i\end{pmatrix}\right)\right]_\phi \tag{S76}$$

$$= -\left[-\dot T_{t_{\rm sp}-t_i}\begin{pmatrix}\phi(t_i)\\ I(t_i^+)\end{pmatrix} + DT_{t_{\rm sp}-t_i}\begin{pmatrix}\phi(t_i)\\ I(t_i^+)\end{pmatrix}\cdot\left(\dot T_{t_i-t_1}\begin{pmatrix}\phi(t_1)\\ I(t_1)\end{pmatrix}\right)\right]_\phi \tag{S77}$$

$$= 1 - \left[DT_{t_{\rm sp}-t_i}\begin{pmatrix}\phi(t_i)\\ I(t_i^+)\end{pmatrix}\cdot f\begin{pmatrix}\phi(t_i)\\ I(t_i^-)\end{pmatrix}\right]_\phi, \tag{S78}$$

where we used in the last line that the ϕ-component of f is 1 at a spike time, Eq. (S12). For $t_i \nearrow t_{i,0}$, also $t_{\rm sp}$ tends to $t_{i,0}$, such that $DT_{t_{\rm sp}-t_i}\begin{pmatrix}\phi(t_i)\\ I(t_i^+)\end{pmatrix}$ becomes the identity matrix and the ϕ-component of $f\begin{pmatrix}\phi(t_i)\\ I(t_i^-)\end{pmatrix}$ tends to its value at a spike time, 1, since t_i tends to a spike time of the dynamics. It follows that

$$\lim_{t_i \nearrow t_{i,0}} \frac{\partial t_{\mathrm{sp}}}{\partial t_i} = 1 - 1 = 0. \tag{S79}$$

This shows that t_{sp} is a smooth function of t_i also if t_i crosses t_{sp}.

III. PSEUDOSPIKE TIME SMOOTHNESS AND CONTINUITY FOR QIF NEURONS WITH EXTENDED COUPLING

A. First type of pseudospikes

In this section, we prove that the pseudospike times for QIF neurons with extended coupling of the first type (Sec. I B 1), including their transitions to ordinary spike times, are continuous and mostly smooth in the network parameters (weights and input spike times). This is condition (i) of main text Sec. III.

We first note that a pseudospike time t_{ps} is smooth in case no network spike crosses the trial end. In this case, the network state at the trial end, i.e. potentials and currents at T, vary smoothly (Sec. II). Since t_{ps} depends on the final network state via smooth functions (Sec. I B 1), it also varies smoothly. Thus, we only need to consider cases where a network spike crosses the trial end. As before, we only consider cases where only one spike crosses the trial end at a time.

1. The spike crosses the trial end

We first show that the spike time changes smoothly with the network parameters, if a pseudospike becomes an ordinary spike or vice versa. Specifically, we consider the case where the kth spike crosses the trial end due to a small, continuous change of a network parameter. This means that at a critical value of this parameter, an ordinary spike (dis-)appears. If the parameter approaches the critical value from one side, the kth ordinary spike shifts towards the trial end, $t_{\text{sp}} \nearrow T$. This implies $V(T) \to -\infty$, since also the voltage reset following t_{sp} shifts towards the trial end from below. When approaching the critical parameter value from the other side, the spike and thus the voltage reset does not happen within the trial, but we have $V(T) \to \infty$, since the neuron comes closer to emitting its kth spike within the trial. In this case, the time of the kth spike, which is a pseudospike, is given by Eq. (S25) with $n_{\text{trial}} = k-1$,

$$t_{\text{sp}} = T + \phi_{\Theta,I_{\text{ps}}} - \Phi_{I_{\text{ps}}}(V(T)). \tag{S80}$$

Because of $\lim_{V(T)\to\infty} \Phi_{I_{\text{ps}}}(V(T)) = \phi_{\Theta,I_{\text{ps}}}$, $\lim_{V(T)\to\infty} t_{\text{sp}} = T$. Thus, the pseudospike (dis-)appears at the trial end, where also the new ordinary spike (dis-)appears. This shows the continuity of the time of the kth spike in case it transitions from being an ordinary spike to being a pseudospike and vice versa.

To show that also the gradient is continuous, we consider a region in parameter space around the critical value for which, if the kth spike is a pseudospike, $V(T)$ is so large that the neuron would emit its kth spike if the trial would not end. We denote the time of this hypothetical ordinary spike by t_{ord}, independent of the spike being before or after T. As established in Sec. II, t_{ord} depends smoothly on the parameters. In particular, the value of its gradient at the transition is equal to its limit taken from either direction. It can be computed using Eqs. (S3), (S9) and (S10) with $V_0 = V(T)$ in case $t_{\text{ord}} \gtrsim T$. The derivatives of t_{ord} with respect to $V(T)$ and the input current at the trial end as well as the derivatives of t_{ps} with respect to $V(T)$ and I_{ps} go to 0 when $V(T) \to \infty$. The derivative of $V(T)$ with respect to the varied parameter simultaneously diverges, however. Since the derivatives of t_{ord} and t_{ps} with respect to $V(T)$ agree in leading order,

$$\frac{\partial t_{\text{ps}}}{\partial V(T)} = -\frac{\partial \Phi_{I_{\text{ps}}(T)}(V(T))}{\partial V(T)} = \frac{-1}{g(I_{\text{ps}}) + (V(T)-1/2)^2} \underset{V(T)\to\infty}{\simeq} -\frac{1}{V(T)^2}, \tag{S81}$$

$$\frac{\partial t_{\text{ord}}}{\partial V(T)} \underset{V(T)\to\infty}{\simeq} -\frac{1}{V(T)^2}, \tag{S82}$$

also the gradients of t_{ord} and t_{ps} asymptotically agree. Hence, the spike time gradient is continuous if the spike time crosses the trial end.

2. A previous spike crosses the trial end

We now show that the spike time of a pseudospike changes smoothly with the network parameters, if a previous output spike of the same neuron crosses the trial end. Specifically, we consider the case where not the kth spike crosses the trial end but the lth spike, where $l < k$. When approaching the transition from the side where the lth spike is an ordinary spike, $n_{\text{trial}} = l$ and $V(T) \to -\infty$. When approaching it from the other side, $n_{\text{trial}} = l - 1$ and

$V(T) \to \infty$. In the former case, we have

$$t_{\mathrm{ps}} = T + (k-l)\phi_{\Theta,I_{\mathrm{ps}}} - \Phi_{I_{\mathrm{ps}}}(V(T)) \underset{V(T)\to-\infty}{\longrightarrow} T + (k-l)\phi_{\Theta,I_{\mathrm{ps}}}, \tag{S83}$$

because $\lim\limits_{V(T)\to-\infty} \Phi_{I_{\mathrm{ps}}}(V(T)) = 0$. In the latter case, we have

$$t_{\mathrm{ps}} = T + (k-(l-1))\phi_{\Theta,I_{\mathrm{ps}}} - \Phi_{I_{\mathrm{ps}}}(V(T)) \underset{V(T)\to\infty}{\longrightarrow} T + (k-l)\phi_{\Theta,I_{\mathrm{ps}}}, \tag{S84}$$

because $\lim\limits_{V(T)\to\infty} \Phi_{I_{\mathrm{ps}}}(V(T)) = \phi_{\Theta,I_{\mathrm{ps}}}$. Thus, t_{ps} is continuous.

Since, in both cases,

$$\frac{\partial t_{\mathrm{ps}}}{\partial I_{\mathrm{ps}}} \to (k-l)\frac{-\pi}{2g^{(3/2)}(I_{\mathrm{ps}})}\frac{\partial g(I_{\mathrm{ps}})}{\partial I_{\mathrm{ps}}} \tag{S85}$$

and

$$\frac{\partial t_{\mathrm{ps}}}{\partial V(T)} = -\frac{\partial \Phi_{I_{\mathrm{ps}}}(V(T))}{\partial V(T)} \simeq -\frac{1}{V(T)^2} \tag{S86}$$

in leading order, also the gradient of t_{ps} is continuous.

3. An input spike crosses the trial end

Finally we show that the spike time of a pseudospike changes continuously with the network parameters, if a spike of another neuron in the network crosses the trial end. Specifically, we first assume that we move along a curve in parameter space that crosses a critical value where an input spike k_{j_0} of neuron j_0 crosses the trial end. Since $V(T)$ changes continuously during the transition, we focus on I_{ps}. When approaching the transition from the side where the k_{j_0}th spike of neuron j_0 is an ordinary spike, $I(T) \to I_{\cancel{k_{j_0}}}(T) + w_{j_0}$, where $I_{\cancel{k_{j_0}}}(T)$ is the value of the input current at the trial end without the effect of spike k_{j_0}. Furthermore Eq. (S27) implies $r_{j_0} \to 0$, because $V_{j_0} \searrow -\infty$. Thus, we have

$$I_{\mathrm{ps}} = I(T) + \sum_j w_j r_j \longrightarrow I_{\cancel{k_{j_0}}}(T) + w_{j_0} + \sum_{j\neq j_0} w_j r_j\,. \tag{S87}$$

When approaching the transition from the other side, $I(T) \to I_{\cancel{k_{j_0}}}(T)$ and $r_{j_0} \to 1$. Thus, we have

$$I_{\mathrm{ps}} = I(T) + \sum_j w_j r_j \longrightarrow I_{\cancel{k_{j_0}}}(T) + \sum_{j\neq j_0} w_j r_j + w_{j_0}\,. \tag{S88}$$

Since both limits agree, t_{ps} is continuous at the transition. The continuity in case neuron j_0 is not directly presynaptic to neuron i is then also guaranteed, since pseudospike times depend continuously on presynaptic pseudospike times.

The gradient of t_{ps} is, however, not continuous in case an input spike crosses the trial end.

IV. GRADIENT STATISTICS OF QIF NEURONS WITH EXTENDED COUPLING

In the following we numerically estimate magnitudes of the gradients that occur in QIF neurons with extended coupling. The neurons receive a high-frequency Poisson input spike train with normally distributed input weights. Inhibitory and excitatory spike inputs balance each other, such that the average input is zero. After a period of equilibration, a test input is provided. We compute the gradient with respect to the test input strength for different realizations of the input spike train and at different test input strengths. To cover the influence of temporal distance the obtained gradients are sorted according to the timing of the spike and presented in different histograms in Fig. S6. Specifically, we bin time beyond the input into five bins of duration 2 (two times the membrane time constant). The gradient of the time of a spike falling in bin number n then contributes to the nth histogram (roman numerals in Fig. S6). The mth bar in this histogram shows the empirical probability that in a single trial (with a randomly chosen test input weight and set of Poisson inputs) a spike time occurs in the nth time bin after the input and that it has a gradient that falls into the mth gradient size bin. The sum over these probabilities is the expected number of spikes per trial.

We observe that gradients of temporally close and of most distant spike times are often smaller than those of spikes with intermediate distance (compare histograms I,V with II,III,IV). This is because inputs usually have little impact on very close and very distant states. However, if a new spike (dis-)appears due to changes in the test input weight, this happens at the trial end, i.e. with maximal temporal distance. These spikes have high sensitivity to the test weight as in the case without further inputs, cf. the larger negative gradient around w_{min} in main text Fig. 1 left, which extends further for longer trial duration. Therefore the largest negative gradients occur in large temporal distance, Fig. S6a IV and V. We find that both lower input variance and the addition of an oscillatory drive reduce the occurring gradients, Fig. S6b. We finally note that for the standard exponential integrate-and-fire neuron with its steep upstroke towards spiking, we observe excessively large gradients already in very short trials.

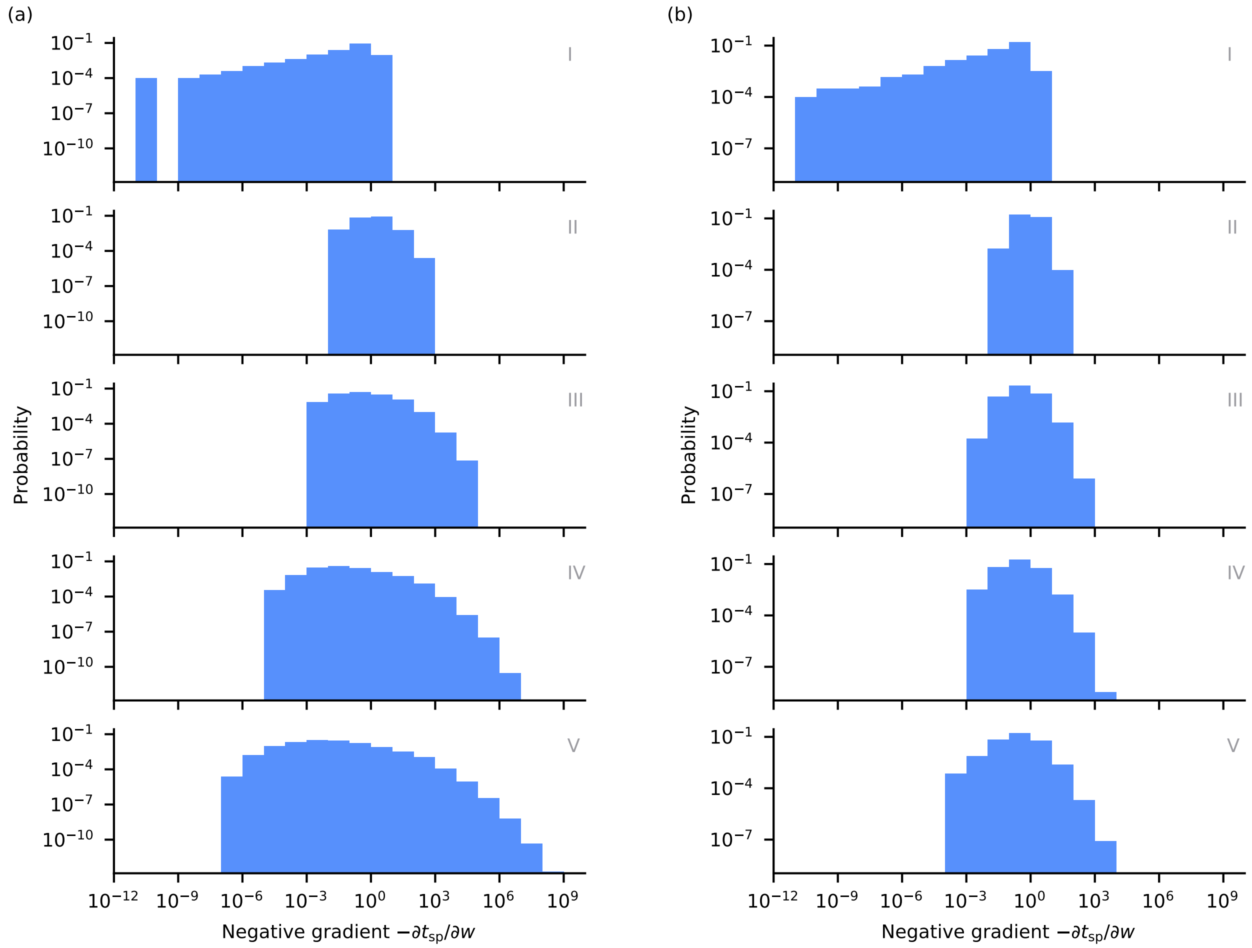

Figure S6. Spike time gradients of QIF neurons with extended coupling. (a) shows results for our standard and (b) for an intrinsically oscillating QIF neuron, which moreover has lower input variance. We compute the gradients with respect to a test input weight and sample them according to the temporal distance of their underlying spike time to the test input (histograms I-V).

V. FURTHER SIMULATION RESULTS

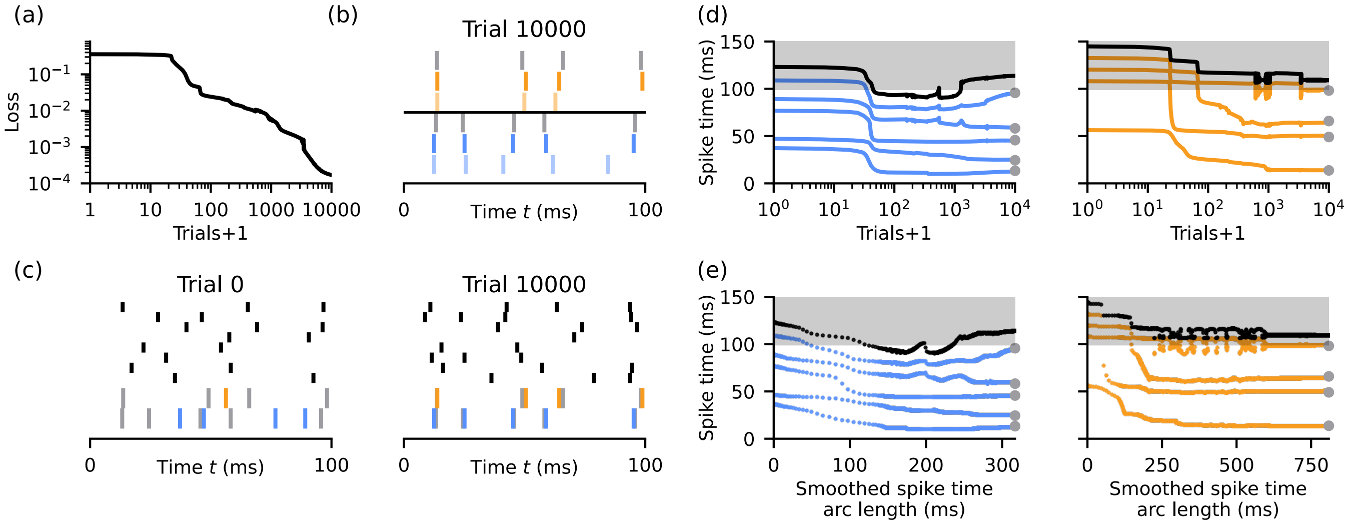

Figure S7. Further results on the learning of precise spikes in an RNN. (Same simulation as shown in main text Fig. 3.) (a) Loss dynamics during learning. (b) Comparison of target spike times (gray), spike times after learning (saturated colors) and spike times after learning if the weights of recurrent connections not targeting the first two neurons are set to 0 after learning (pale colors). The partially large deviations of the latter spike times from the targets illustrate that learning utilizes recurrent connections not involving the target neurons. (c) (Reproduced from main text Fig. 3b.) Left: Spikes of network neurons before learning. Spikes of the first two neurons are colored, their target times are displayed in gray. Right: Learning changes the network dynamics such that the first two neurons spike precisely at the desired values (the colored spikes mostly cover the gray ones). (d) Left: (Reproduced from main text Fig. 3c.) Evolution of the spike times of the first neuron during learning. The times of the spikes that are supposed to lie within the trial (blue traces) shift towards their target values (gray circles). The next spike (black trace) is supposed to lie outside the trial. It occurs transiently within the trial but becomes a pseudospike towards the end of learning again, as required. Gray area indicates pseudospikes. Right: Same as left but for the second neuron. (e) Same as (d) but the spike times are shown as a function of the arc length of the smoothed spike time trajectory. The spike times of the first neuron change continuously. The spike times of the second neuron exhibit jumps at which the times of later spikes shift to the times of earlier spikes at the previous trial. This is because of highly localized large gradients and can be avoided by using variable learning rates (see Fig. S8). Furthermore, the spike times exhibit oscillations after the initial large shifts.

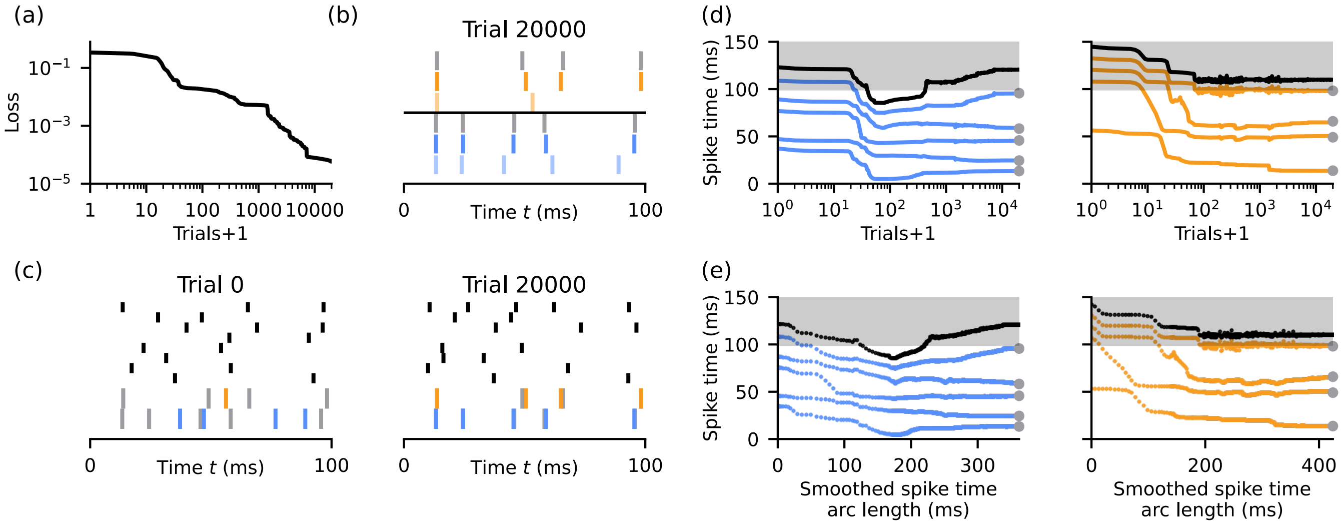

Figure S8. Same as Fig. S7 but using an alternative optimization method, which restricts the maximal step size (see Tab. S6). It results in continuous spike time changes (d,e), which reflects that the occurring gradients are large but finite.

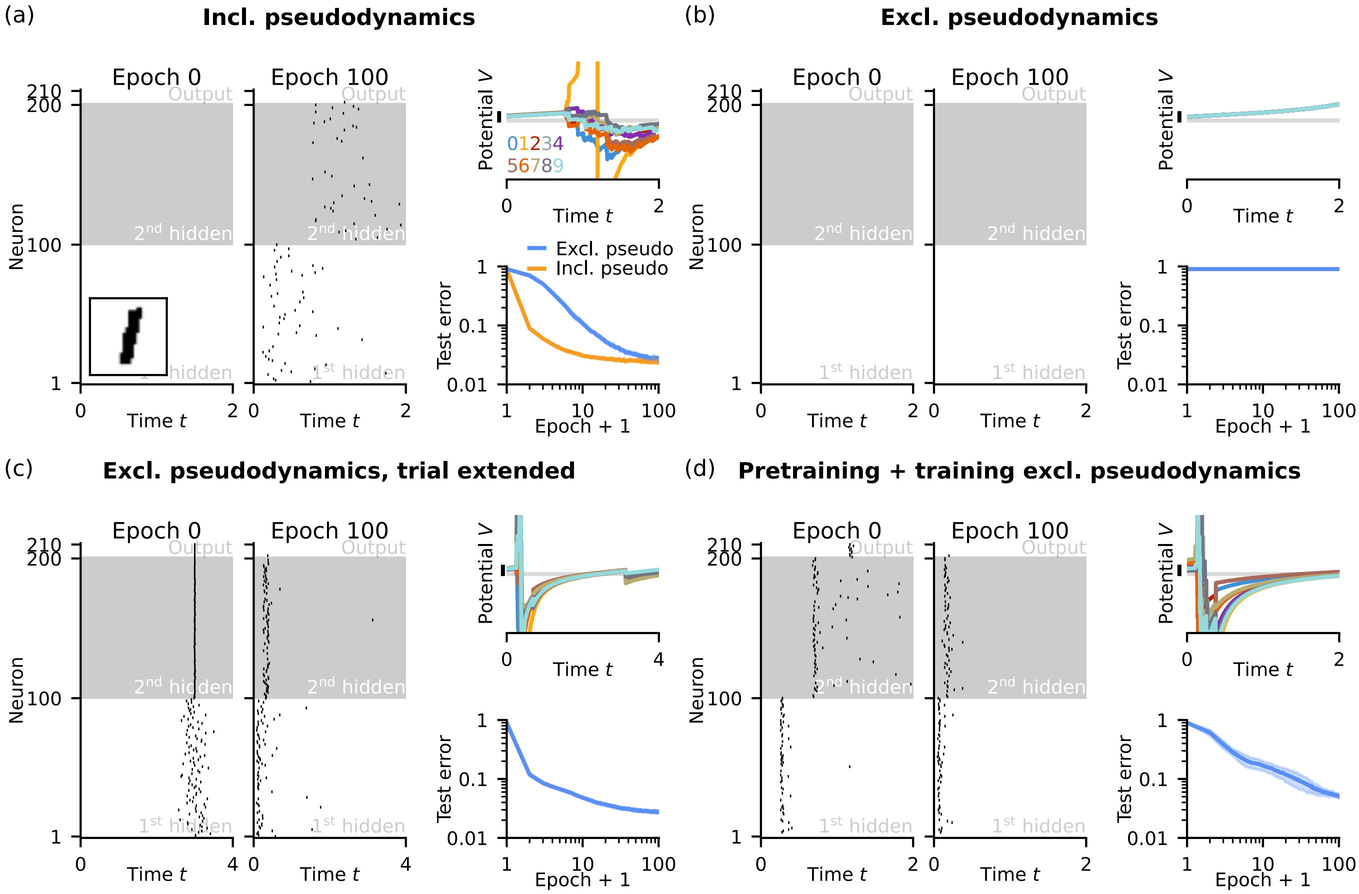

Figure S9. Necessity of pseudodynamics for the learning of the MNIST task. The panels show different learning variants. Each panel is structured like main text Fig. 4. The spike trains and voltage traces are evoked by the same stimulus, which is displayed as inset in (a). (a) (Reproduced from main text Fig. 4.) Learning including pseudodynamics. (b) Learning excluding pseudodynamics. No new spikes can be added, hence learning is unsuccessful. (c) Learning excluding pseudodynamics but with an extended trial duration. Due to the extended trial and the oscillatory neuron model, all neurons already spike before learning, enabling successful learning. (d) Learning excluding pseudodynamics after pretraining. During pretraining, pseudodynamics are used to shift output neuron spikes inside the trial using an appropriate loss (see Tab. S7). This loss is agnostic to the final task. After pretraining, the network neurons spike sufficiently often (leftmost panel) to enable successful learning. To compute the loss in panels (b-d), spike times of output neurons that do not spike within the trial are set to the trial duration T.

VI. SUPPLEMENTARY TABLES

Table S1. Analysis of network behavior for the MNIST-task on the test set. Only spikes that lie within the trial and are relevant for the classification, i.e. that happen before the first output spike, are considered. Specifically, for the computation of the accuracy, only ordinary output spikes are considered valid. For the loss, the spike times of output neurons that do not spike within the trial are set to T; this leads to a large loss also after training. A hidden neuron is considered silent if it does not spike before the first output spike (or, if there is no ordinary output spike, within the trial) for any input image. Similarly, the activity is the number of ordinary, hidden layer spikes before the first output spike (or, if there is no ordinary output spike, within the trial) per hidden neuron. Values represent mean ± std over ten network instances, clipped to lie within the possible ranges, where necessary. The standard deviation is zero for the accuracy and the loss before learning because there are no output spikes at all.

	Before learning	After learning
Accuracy	$9.8\,\%$	$(97.3 \pm 0.3)\,\%$
Loss	2.320	1.686 ± 0.007
Silent neurons	$(99.8^{+0.2}_{-0.3})\%$	$(0.1^{+0.3}_{-0.1})\%$
Activity	$(1^{+2}_{-1}) \times 10^{-6}$	$(31.6 \pm 0.1) \times 10^{-2}$

Table S2. Same as Tab. S1 but including the use of pseudospikes for classification and considering not only spikes before the first output spike but all ordinary spikes for the computation of the fraction of silent neurons and the activity.

	Before learning	After learning
Accuracy	$(10.4 \pm 0.7)\,\%$	$(97.6 \pm 0.2)\,\%$
Loss	2.345 ± 0.004	0.139 ± 0.012
Silent neurons	$(99.8^{+0.2}_{-0.3})\%$	$0\,\%$
Activity	$(1^{+2}_{-1}) \times 10^{-6}$	$(41.6 \pm 1.2) \times 10^{-2}$

VII. SUPPLEMENTARY DISCUSSION

Our method and networks, in particular the one we use for the MNIST task, possess notable connections and parallels to standard and recent machine learning techniques. Probably the most direct connection is the fact that the working mode of our networks during the pseudodynamics of the first type corresponds to the functioning of a standard rate neural network. This means that during the pseudodynamics neurons interact via weighted sums of rate-like quantities, as shown in Secs. I B 1 and I B 3. A similar mapping can, however, not be obtained for the ordinary dynamics, even if there is at most one spike per neuron. This is because an ordinary spike time does not depend on the (possibly nonlinearly transformed) ordinary presynaptic spike times via their linear superposition, due to the nonlinear single neuron dynamics. Thus, besides viewing our networks as fully spiking, one can also interpret them as a hybrid network, working in a spiking mode during the ordinary dynamics and in a rate mode during the pseudodynamics. We note that there are mappings between networks of ReLUs and networks of carefully constructed non-leaky integrate-and-fire neurons that have step-like synaptic input currents and spike only once [34, 35].

An important feature of our networks is that after learning, at inference time, it suffices to keep the ordinary dynamics only to achieve the tasks. In the setup that we use for the MNIST task, neurons typically only generate ordinary spikes in response to a subset of input images. Such sparsity is also important in machine learning contexts [36]. It is a form of ephemeral (per example) sparsity, analogous to the sparsity induced by the ReLU activation function, which outputs zero for any negative (subthreshold) input, such that only an input-dependent subset of neurons is active for each input. During learning of the MNIST task, the interactions between pseudospikes imply that all neuron weights and activities affect the higher layer dynamics. This bears a similarity to using an activation function that does not clamp input ranges to zero, such as leaky ReLU or a related smooth function, in a non-spiking network during learning. The removal of the pseudodynamics after learning then resembles the replacement of this activation function by a ReLU function. This has been studied in [37] in the context of large language models: Like in our networks the replacement sparsifies the network activity. It, however, also requires brief additional training. This is not the case in our networks, since pseudospikes do not affect ordinary ones and the ordinary spikes alone solve the task after training.

Pseudospikes allow the gradient to see behind the trial end. This bears some similarity to the property of a surrogate gradient to see under the threshold [38]. In particular, both approaches can foresee whether spikes are about to appear for certain weight changes. However, in contrast to surrogate gradient descent, where the nonlinearities change, in our approach the computation of the dynamics (forward path) and the computation of the gradient (backward path) use the same dynamical equations. Further the contributions of the gradients of the ordinary spike times to the total gradient are not affected by the presence of pseudospikes.

A possible future research direction is the implementation of our approach on neuromorphic hardware. For this, it may be useful that with appropriately initialized network and trial parameters or after appropriate pretraining using pseudodynamics, training networks without pseudodynamics can be possible (Fig. S9). If they are required, the pseudodynamics need special consideration depending on the learning scenario [39]. This holds in particular for the pseudodynamics of the first type. They need at one point in time, namely at the ordinary trial end, the transmission of nonlinearly transformed subthreshold potentials between all connected neurons and the nonlinear computation of the strength of the constant input currents. In the case of on-chip learning, this may necessitate reading out the network state at T and newly setting up the network with the appropriate currents and no interactions. The implementation of the second type of pseudodynamics may be easier: these dynamics first continue according to the same rules as the ordinary ones and at some fixed time point switch to different, fixed dynamics. In the case of off-chip learning, i.e. learning on conventional hardware and subsequent deployment of the learned network on neuromorphic hardware, the pseudodynamics do not cause a problem as they are not used after learning. If additional finetuning is required after deployment, adding spikes and thus pseudodynamics need not be necessary. In the case of chip-in-the-loop learning, i.e. neuromorphic hardware is used for the forward run and conventional hardware for the weight update computation, the pseudodynamics could be computed on the conventional hardware as well.

VIII. MODEL AND TASK DETAILS

This section provides further details on the figures presented in our article. The formatting mostly follows ref. [40]. If not noted otherwise, the initial conditions are $V(0) = 0$ (or the corresponding phase) and $I(0) = 0$.

Table S3. Description of the QIF neuron model of main text Fig. 1.

A	**Model summary**	
Population	A single neuron	
Neuron	QIF	
Synapse	Extended coupling (exponentially decaying input current)	
Input	One or two input spikes	
B	**Neuron and synapse model**	
Name	QIF neuron with extended coupling	
Neuron dynamics	$\dot{V}(t) = V(t)(V(t) - 1) + I(t)$	(Subthreshold dynamics)
	$V_\Theta = \infty$	(Threshold)
	$V_{\text{reset}} = -\infty$	(Reset)
Synaptic dynamics	$\tau_\text{s}\dot{I}(t) = -I(t) + \tau_\text{s} w \delta(t - t_\text{e})$	(One input)
	$\tau_\text{s}\dot{I}(t) = -I(t) + \tau_\text{s} w_\text{e} \delta(t - t_\text{e}) + \tau_\text{s} w_\text{i} \delta(t - t_\text{i})$	(Two inputs)
C	**Input**	
Type	**Description**	
One input	A single excitatory input with varying weight w	
Two inputs	An excitatory input and an inhibitory input with varying time t_i	

D		**Parameters**
Parameter	**Value**	**Description**
T	4	Trial length
τ_s	1/2	Synaptic time constant
$w_{\min}$	2.47	Minimal weight necessary to elicit a spike at infinity
t_e	0.5	Time of excitatory input in both input cases
w_e	$1.5 w_{\min}$	Weight of excitatory input in case of two inputs
w_i	$-w_{\min}$	Weight of inhibitory input in case of two inputs

Table S4. Description of the LIF neuron model of main text Fig. 1.

A	**Model summary**
Population	A single neuron
Neuron	LIF
Synapse	Extended coupling (exponentially decaying input current)
Input	One or two input spikes

B	**Neuron and synapse model**	
Name	LIF neuron with extended coupling	
Neuron dynamics	$\dot{V}(t) = -V(t) + I(t)$	(Subthreshold dynamics)
	$V_\Theta = 1$	(Threshold)
	$V_{\mathrm{reset}} = 0$	(Reset)
Synaptic dynamics	$\tau_\mathrm{s}\dot{I}(t) = -I(t) + \tau_\mathrm{s} w \delta(t - t_\mathrm{e})$	(One input)
	$\tau_\mathrm{s}\dot{I}(t) = -I(t) + \tau_\mathrm{s} w_\mathrm{e} \delta(t - t_\mathrm{e}) + \tau_\mathrm{s} w_\mathrm{i} \delta(t - t_\mathrm{i})$	(Two inputs)

C	**Input**
Type	**Description**
One input	A single excitatory input with varying weight w
Two inputs	An excitatory input and an inhibitory input with varying time t_i

D		**Parameters**
Parameter	**Value**	**Description**
T	3	Trial length
τ_s	1/2	Synaptic time constant
w_{min}	4	Minimal weight necessary to elicit a spike
t_e	0.5	Time of excitatory input in both input cases
w_e	$1.4 w_{\mathrm{min}}$	Weight of excitatory input in case of two inputs
w_i	$-w_{\mathrm{min}}$	Weight of inhibitory input in case of two inputs

Table S5. Description of the QIF neuron model of main text Fig. 2.

A	**Model summary**
Population	A single neuron
Neuron	QIF
Synapse	Extended coupling (exponentially decaying input current)
Input	Combination of fixed, random as well as learnable input spikes
Learning	Gradient descent on first two spike times

B	**Neuron and synapse model**
Name	QIF neuron with extended coupling
Neuron dynamics	$\dot{V}(t) = V(t)(V(t)-1) + I(t)$ (Subthreshold dynamics)
	$V_\Theta = \infty$ (Threshold)
	$V_{\mathrm{reset}} = -\infty$ (Reset)
Synaptic dynamics	$\tau_{\mathrm{s}}\dot{I}(t) = -I(t) + \tau_{\mathrm{s}} \sum_{j=1}^{10} w_j^{\mathrm{fix}} \delta(t - t_j^{\mathrm{fix}}) + \tau_{\mathrm{s}} \sum_{j=1}^{2} w_j^{\mathrm{learn}} \delta(t - t_j^{\mathrm{learn}})$
Pseudodynamics	After the trial end, neurons evolve as described in Sec. I B 1

C	**Input**
Fixed inputs	Twenty input spikes, times t_j^{fix} randomly drawn from uniform distribution, weights w_j^{fix} randomly drawn from normal distribution with mean 0 and variance 1
Learnable inputs	Two input spikes, times t_j^{learn} are initially 1 and 9, weights w_j^{learn} are initially 0

D	**Learning**
Loss description	Mean squared error loss
Loss function	$L(p) = \frac{1}{2} \sum_{k=1}^{2} \left(t_k(p) - t_k^{\mathrm{tar}}\right)^2$ ($t_k(p)$ denotes the kth output spike)
Learnable parameters p	Times t_j^{learn} and weights w_j^{learn} of the learnable input spikes
Optimization method	Gradient descent with element-wise gradient clipping at 2.2×10^{-2}

E		**Parameters**
Parameter	**Value**	**Description**
T	10	Trial length
τ_{s}	1/2	Synaptic time constant
η	0.1	Learning rate
t_k^{tar}	$\{2.5, 7.5\}$	Target spike times
N_{trial}	3000	Number of trials

Table S6. Description of the RNN of main text Fig. 3.

A	**Model summary**
Population	One population
Connectivity	All-to-all
Neuron	QIF
Synapse	Extended coupling (exponentially decaying input current)
Input	Excitatory and inhibitory Poisson spike trains
Learning	Gradient descent on spike times of two network neurons
B	**Population**
One population of N QIF neurons.	
C	**Connectivity**
All-to-all recurrent connectivity, weights from neuron j to neuron i denoted with w_{ij}, weights initially set to 0	
D	**Neuron and synapse model**
Name	QIF neuron with extended coupling
Neuron dynamics	$\dot{V}_i(t) = V_i(t)(V_i(t) - 1) + I_i(t)$ (Subthreshold dynamics of neuron i) $V_\Theta = \infty$ (Threshold) $V_{\text{reset}} = -\infty$ (Reset)
Synaptic dynamics	$\tau_{\text{s}} \dot{I}_i(t) = -I_i(t) + \tau_{\text{s}} w_{\text{e}}^{\text{in}} S_{\text{e},i}(t) + \tau_{\text{s}} w_{\text{i}}^{\text{in}} S_{\text{i},i}(t) + \tau_{\text{s}} \sum_{j=1, j\neq i}^{N} w_{ij} \sum_{k_j} \delta(t - t_{k_j})$ (t_{k_j}: kth spike time of neuron j)
Pseudodynamics	After the trial end, neurons evolve as described in Sec. I B 1
E	**Input**
Each neuron independently receives one excitatory Poisson spike train $S_{\text{e},i}(t) = \sum_k \delta(t - t_k)$ with fixed weight w_{e}^{in} and one inhibitory Poisson spike train $S_{\text{i},i}(t) = \sum_k \delta(t - t_k)$ with fixed weight w_{i}^{in}. Both have the same rate r_{in}.	
F	**Learning**
Loss description	Weighted mean squared error loss
Loss function	$L(p) = \frac{1}{N_{\text{tar}}} \sum_{i=1}^{N_{\text{tar}}} \sum_{k_i=1}^{N_{\text{tar},i}} \left(\frac{t_{k_i}(p) - t_{k_i}^{\text{tar}}}{t_{k_i}^{\text{tar}} + 2} \right)^2 \left(1 - \delta_{k_i N_{\text{tar},i}} H(t_{N_{\text{tar},i}}^{\text{tar}} - t_{k_i}(p)) \right)$ (H is the Heaviside step function)
Learnable parameters p	Initial states $V_i(0)$, $I_i(0)$ and recurrent weights w_{ij}
Target times	For N_{tar} out of the N network neurons, target times $t_{k_i}^{\text{tar}}$ are drawn from a Poisson process with rate r_i^{tar} and absolute refractoriness 1. In addition to these $N_{\text{tar},i} - 1$ target spikes, a further target time $t_{N_{\text{tar},i}}^{\text{tar}} = 1.1T$ is used to avoid having more spikes than wanted within the trial.
Optimization method	AdaBelief [41] with exponential learning rate decay
Alternative optimization method	AdaBelief [41], but with variable learning rate. In every step, the weight update is computed for a set of learning rates. Of all weight updates with a resulting maximal spike time change of less than 0.5, the one resulting in the smallest error is selected.

Table S6. (continued)

G		**Parameters**
Parameter	**Value**	**Description**
T	10	Trial length
N	10	Number of neurons
τ_{s}	1/2	Synaptic time constant
$w_{\mathrm{e}}^{\mathrm{in}}$	5	Excitatory input weight
$w_{\mathrm{i}}^{\mathrm{in}}$	$-w_{\mathrm{e}}$	Inhibitory input weight
r_{in}	1	Input rate
r_1^{tar}	1	Rate of the Poisson process used to generate target times for the first target neuron
r_2^{tar}	1/2	Rate of the Poisson process used to generate target times for the second target neuron
N_{tar}	2	Number of neurons whose spike times are learned
η	0.01	Learning rate
τ_η	2×10^3	Time scale of exponential learning rate decay (not used for the alternative optimization method)
	10^{-5}–10^2	Range of the 50 possible learning rates, evenly distributed in log-space, that are used in the alternative optimization method
β_1	0.9	Exponential decay rate used to track first moment of gradient in AdaBelief
β_2	0.999	Exponential decay rate used to track second moment of gradient in AdaBelief
	0.99	β_2 in the case of the alternative optimization method
N_{trial}	10 000	Number of trials
	20 000	N_{trial} in the case of the alternative optimization method

Table S7. Description of the multi-layer network of main text Fig. 4 and Fig. S9.

A	**Model summary**
Population	Three: two hidden layers, one output layer
Connectivity	Feedforward connectivity only
Neuron	Oscillatory QIF
Synapse	Infinitesimally short coupling (delta-pulse coupling)
Input	Binarized MNIST images encoded with single spike per pixel
Learning	Gradient descent learning of time-to-first spike encoded image label
B	**Population**
Input layer	One input layer consisting of $N^{(0)}$ neurons with fixed spike times
Hidden layers	Two hidden layers consisting of $N_\mathrm{h} = N^{(1)} = N^{(2)}$ neurons each
Output layer	One output layer consisting of $N^{(3)} = N_\mathrm{tar} = 10$ neurons, one for each label
C	**Connectivity**
Full feedforward connectivity between subsequent layers, no recurrent connections, weight from neuron j in layer $l-1$ to neuron i in layer l denoted with $w_{ij}^{(l)}$, weights initially randomly drawn from uniform distribution	
D	**Neuron and synapse model**
Name	Oscillatory QIF neuron with delta-pulse coupling
Neuron dynamics	$\tau_\mathrm{m}\dot{V}_i^{(l)}(t) = V_i^{(l)}(t)(V_i^{(l)}(t)-1) + I_i^{(l)}(t)$ (Subthreshold dynamics of neuron i in layer l)
	$V_\Theta = \infty$ (Threshold)
	$V_\mathrm{reset} = -\infty$ (Reset)
Synaptic dynamics	$I_i^{(l)}(t) = I_0 + \tau_\mathrm{m} \sum_{j=1}^{N^{(l-1)}} w_{ij}^{(l)} \sum_{k_j} \delta(t - t_{k_j})$ (t_{k_j}: kth spike time of neuron j)
Neuron dynamics (angle space)	$\dot{\phi}_i^{(l)}(t) = 1$ (Between spikes)
	$\phi_\Theta = \tau_\mathrm{m}\pi/\sqrt{I_0 - \frac{1}{4}}$ (Threshold)
	$\phi_\mathrm{reset} = 0$ (Reset)
Synaptic dynamics (angle space)	$\phi_i^{(l)}(t_{k_j}^+) = H_{w_{ij}^{(l)}}(\phi_i^{(l)}(t_{k_j}^-)) = \Phi(\Phi^{-1}(\phi_i^{(l)}(t_{k_j}^-)) + w_{ij}^{(l)})$ (t_{k_j} denotes the kth spike of neuron j)
Pseudodynamics	After the trial end, neurons evolve as described in Sec. I B 3
E	**Input**
Pixel values are binarized, input neurons corresponding to active pixels spike once at 0.02, others do not spike at all	

Table S7. (continued)

F	**Learning**
Loss description	Cross-entropy loss on first spike times of the output neurons, regularization term to encourage early spiking [42]
Loss function (single input)	$L(p) = \sum_{i=1}^{N_\mathrm{tar}} y_{\mathrm{tar},i} \log(y_i(p)) + \gamma \sum_{i=1}^{N_\mathrm{tar}} y_{\mathrm{tar},i} \left(\exp(t_i^{(3)}(p)/T) - 1\right)$ $y_i(p) = \frac{\exp(-t_i^{(3)}(p))}{\sum_j \exp(-t_j^{(3)}(p))}$ (softmax) $y_{\mathrm{tar},i} = \delta_{i,(\mathrm{label}+1)}$ (one-hot encoded target label)
Loss function (pretraining in Fig. S9(d))	$L(p) = \frac{1}{N_\mathrm{tar}} \sum_{i=1}^{N_\mathrm{tar}} \left(t_i^{(3)} - T\right)^2 H\left(t_i^{(3)} - T\right)$ (H is the Heaviside step function)
Learnable parameters p	Initial states $V_i^{(l)}(0)$ and feedforward weights $w_{ij}^{(l)}$
Mini batches	Batches of size N_batch are used, loss is averaged over batch
Optimization method	AdaBelief [41] with exponential learning rate decay
Input regularization	To avoid overfitting, the state of each binarized pixel is flipped with probability p_flip during learning
Hyperparameter search and evaluation	Training data set: 55000 images, validation data set: 5000 images, test data set: 10000, hyperparameters are manually tuned using the validation data set, network performance is evaluated on held-out test data set

G		**Parameters**
Parameter	**Value**	**Description**
T	2	Trial length
$N^{(0)}$	784	Number of input neurons/pixels
N_h	100	Number of hidden layer neurons
N_tar	10	Number of output neurons
τ_m	$6/\pi$	Membrane time constant
I_0	$5/4$	Constant input current component
	$\mathcal{U}([\frac{-0.5}{\sqrt{N^{(l-1)}}}, \frac{0.5}{\sqrt{N^{(l-1)}}}])$	Distribution of weights $w_{ij}^{(l)}$ before learning
	$\phi_\Theta/2$	Value of initial states $V_i^{(l)}(0)$ before learning
γ	10^{-2}	Regularization parameter
N_batch	1000	Batch size
η	4×10^{-3}	Learning rate
τ_η	10^2	Time scale of exponential learning rate decay
β_1	0.9	Exponential decay rate used to track first moment of gradient in AdaBelief
β_2	0.999	Exponential decay rate used to track second moment of gradient in AdaBelief
p_flip	0.02	Flip probability of each pixel during learning
N_epoch	100	Number of epochs (passes through the entire training data set) used for learning
	1	Number of epochs used for pretraining in Fig. S9(d).

Table S8. Description of the multi-layer network of Fig. S2.

A	**Model summary**
Population	Two: One hidden layer, one output layer
Connectivity	Feedforward connectivity only
Neuron	QIF
Synapse	Extended coupling (exponentially decaying input current)
Input	Two input spikes

B	**Population**
Input layer	One input layer consisting of two neurons with fixed spike times
Hidden layers	One hidden layer consisting of two neurons
Output layer	One output layer consisting of one neuron

C	**Connectivity**

Input neuron 1 excites both hidden neurons with the same variable synaptic strength w, $w^1_{11} = w^1_{21} = w$. Input neuron 2 has no connection to hidden neuron 1, $w^1_{12} = 0$, and inhibits hidden neuron 2 with fixed synaptic strength $w^1_{22} = -2$. Hidden neuron 1 excites the output neuron with fixed synaptic strength $w^2_{11} = 3$. Hidden neuron 2 inhibits the output neuron with fixed synaptic strength $w^2_{12} = -1$. w changes from 2 to 6 in steps of $\Delta w = 10^{-6}$. We compute the spike time gradient with respect to the synaptic weight w using the change of same spike times Δt_{sp} between subsequent w as $\Delta t_{\mathrm{sp}}/\Delta w$.

D	**Neuron and synapse model**
Name	QIF neuron with extended coupling
Neuron dynamics (angle space)	$\dot{\phi}_i^{(l)}(t) = \cos(\pi\phi_i^{(l)}(t))\left(\cos(\pi\phi_i^{(l)}(t)) + \frac{1}{\pi}\sin(\pi\phi_i^{(l)}(t))\right) + \frac{1}{\pi^2}\sin^2(\pi\phi_i^{(l)}(t))I_i^{(l)}(t)$ (Subthreshold dynamics of neuron i in layer l)
	$\phi_\Theta = 1$ (Threshold)
	$\phi_{\mathrm{reset}} = 0$ (Reset)
Synaptic dynamics	$\tau_{\mathrm{s}}\dot{I}_i^{(l)}(t) = -I_i^{(l)}(t) + \tau_{\mathrm{s}}\sum_{j=1}^{2} w_{ij}^{(l)} \sum_{k_j}\delta(t - t_{k_j})$ (t_{k_j}: kth spike time of neuron j)
Pseudodynamics	After the trial end, neurons evolve as described in Sec. I B 2

E	**Input**
Type	**Description**
Input spikes	One input spike from input layer neuron 1 at time $t = 1$. One input spike from input layer neuron 2 at time $t = 0$.

F		**Parameters**
Parameter	**Value**	**Description**
T	8	Trial length
d	2	Pseudospike dynamics parameter, Eq. (S31)

Table S9. Description of the QIF neuron model of Fig. S3.

A	**Model summary**
Population	A single neuron
Neuron	QIF
Synapse	Extended coupling (exponentially decaying input current)
Input	One test input

B	**Neuron and synapse model**
Name	QIF neuron with extended coupling
Neuron dynamics (angle space)	$\dot{\phi}(t) = \cos(\pi\phi(t))\big(\cos(\pi\phi(t)) + \frac{1}{\pi}\sin(\pi\phi(t))\big) + \frac{1}{\pi^2}\sin^2(\pi\phi(t))I(t)$ (Subthreshold dynamics of neuron i in layer l)
	$\phi_\Theta = 1$ (Threshold)
	$\phi_{\text{reset}} = 0$ (Reset)
Synaptic dynamics	$\tau_\text{s}\dot{I}(t) = -I(t)$ ($I(0) = w$: weight of test input)

C	**Input**
Type	**Description**
Test input	Input time $t = 0$, input weight w varied between $w_{\text{min}} = -8.5$ and $w_{\text{max}} = 60$ in steps of 10^{-4}.

D		**Parameters**
Parameter	**Value**	**Description**
T	10	Trial length
$\phi(0)$	$\Phi(3) \approx 0.74$	Initial phase

Table S10. Description of the QIF neuron model of Figs. S4 and S5.

A	**Model summary**
Population	A single neuron
Neuron	QIF
Synapse	Extended coupling (exponentially decaying input current)
Input	One test input, four further inputs.

B	**Neuron and synapse model**
Name	QIF neuron with extended coupling
Neuron dynamics (angle space)	$\dot{\phi}(t) = \cos(\pi\phi(t))\big(\cos(\pi\phi(t)) + \frac{1}{\pi}\sin(\pi\phi(t))\big) + \frac{1}{\pi^2}\sin^2(\pi\phi(t))I(t)$ (Subthreshold dynamics of neuron i in layer l)
	$\phi_\Theta = 1$ (Threshold)
	$\phi_{\text{reset}} = 0$ (Reset)
Synaptic dynamics	$\tau_\text{s}\dot{I}(t) = -I(t) + \tau_\text{s}\sum_j w_j\delta(t - t_j)$ (w_j, t_j: weight and time of jth input)

C	**Input**
Type	**Description**
Test input	Fig. S4: Input time t_i varied between 0 and 8 in steps of 10^{-5}, input weight $w_i = -3$.
	Fig. S5: Input time $t_i = 2.22$, input weight w_i varied between $w_{\text{min}} = -8.5$ and $w_{\text{max}} = 60$ in steps of 10^{-4}.
Further inputs	Input from input neuron 1 at times 1 and 1.5, weight 2. Input from input neuron 2 at time 3, weight -3. Input from input neuron 3 at time 5, weight 4.

D		**Parameters**
Parameter	**Value**	**Description**
T	10	Trial length

Table S11. Description of the QIF neuron models and the analysis of Fig. S6.

A	**Model summary**
Population	A single neuron
Neuron	(a) QIF, (b) Oscillatory QIF
Synapse	Extended coupling (exponentially decaying input current)
Input	One test input, balanced Poisson spike train

B	**Neuron and synapse model**	
Name	QIF neuron with extended coupling	
Neuron dynamics	$\dot{V}(t) = V(t)(V(t) - 1) + I(t)$	(Subthreshold dynamics)
	$V_\Theta = 10000$	(Threshold)
	$V_{\text{reset}} = -10000$	(Reset)
Synaptic dynamics	$\tau_{\text{s}}\dot{I}(t) = -I(t) + I_0 + \tau_{\text{s}} \sum_i w_i \delta(t - t_i)$	
		(w_i, t_i: weight and time of ith input spike)

C	**Input**
Type	**Description**
Test input	Input time $t = 5$, uniformly distributed, strength between $w_{\text{min}} = -1.5$ and $w_{\text{max}} = 1.5$.
Poisson input	Input arrivals during the entire trial, frequency 10 (1kHz), weights w_i normally distributed, standard deviation (a) $\sigma = 0.5$, (b) $\sigma = 0.25$.

D		**Parameters**
Parameter	**Value**	**Description**
T	15	Trial length
I_0	(a) 0, (b) 0.5	Constant drive

E	**Analysis**
Type	**Description**
Empirical probability estimation	We consider 10000 sets of trials. In a single set, the randomly chosen Poisson input spike trains are kept the same, while we sample the test input. To resolve also steep gradients, we use an adaptive sampling scheme: The test input weight is decreased from w_{max} to w_{min} with an initial and (in absolute value) maximal step size of $\Delta w = -0.1$. The spike times thus increase between subsequent trials. We choose as desired maximum of the spike time differences Δt_{sp} between same spikes in trial $i + 1$ and i the value 0.1. If it is exceeded by a factor of 2, trial $i + 1$ is discarded and Δw is reduced by a factor of 2. If the observed maximum is smaller than desired by a factor of 2, Δw is increased by a factor of 2, up to the maximal step size. We compute the negative gradients via $-\Delta t_{\text{sp}}/\Delta w$. After the trial set is completed, we sum the lengths of the test weight intervals for which a spike lies in time bin n (bin size 2) and has a gradient in size bin m. The result is normalized by the entire test weight interval sampled. This gives the trial set's probability estimate for bin m in histogram n. Averaging over all trial sets yields the final result.

[1] P. E. Latham, B. J. Richmond, P. G. Nelson, and S. Nirenberg, Intrinsic dynamics in neuronal networks. i. theory, Journal of Neurophysiology **83**, 808 (2000).

[2] E. Izhikevich, *Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting* (MIT Press, Cambridge, 2007).

[3] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal Dynamics - From single neurons to networks and models of cognition* (Cambridge University Press, Cambridge, 2014).

[4] T. P. Vogels, K. Rajan, and L. Abbott, Neural network dynamics, Annual Review of Neuroscience **28**, 357 (2005).

[5] A. Burkitt, A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input, Biol. Cybern. **95**, 1 (2006).

[6] R.-M. Memmesheimer, R. Rubin, B. Ölveczky, and H. Sompolinsky, Learning precisely timed spikes, Neuron **82**, 011053 (2014).

[7] W. R. Inc., Mathematica, Version 13.2, champaign, IL, 2023.

[8] E. Kamke, *Differentialgleichungen. Lösungsmethoden und Lösungen* (Teubner, Stuttgart, 1977).

[9] B. Ermentrout and N. Kopell, Parabolic bursting in an excitable system coupled with a slow oscillation, SIAM J. Appl. Math. **2**, 233 (1986).

[10] H. Tuckwell, *Introduction to theoretical neurobiology: Volume 1. Linear cable theory and dendritic structure* (Cambridge Univ. Press, Cambridge, 1988).

[11] H. Tuckwell, *Introduction to theoretical neurobiology: Volume 2. Nonlinear and stochastic theories* (Cambridge Univ. Press, Cambridge, 1988).

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

[13] R.-M. Memmesheimer and M. Timme, Designing the dynamics of spiking neural networks, Physical Review Letters **97**, 188101 (2006).

[14] R.-M. Memmesheimer, Quantitative prediction of intermittent high-frequency oscillations in neural networks with supralinear dendritic interactions., Proc. Natl Acad. Sci. USA **107**, 11092 (2010).

[15] R. Mirollo and S. Strogatz, Synchronization of pulse coupled biological oscillators, SIAM J. Appl. Math. **50**, 1645 (1990).

[16] R.-M. Memmesheimer and M. Timme, Designing complex networks, Physica D **224**, 182 (2006).

[17] A. Viriyopase, R.-M. Memmesheimer, and S. Gielen, Analyzing the competition of gamma rhythms with delayed pulse-coupled oscillators in phase representation, Phys. Rev. E **98**, 022217 (2018).

[18] M. D'Haene, B. Schrauwen, J. Van Campenhout, and D. Stroobandt, Accelerating Event-Driven Simulation of Spiking Neurons with Multiple Synaptic Time Constants, Neural Computation **21**, 1068 (2009), https://direct.mit.edu/neco/article-pdf/21/4/1068/818994/neco.2008.02-08-707.pdf.

[19] R. Brette, M. Rudolph, T. Carnevale, M. Hines, D. Beeman, J. M. Bower, M. Diesmann, A. Morrison, P. H. Goodman, F. C. Harris, M. Zirpe, T. Natschläger, D. Pecevski, B. Ermentrout, M. Djurfeldt, A. Lansner, O. Rochel, T. Vieville, E. Muller, A. P. Davison, S. El Boustani, and A. Destexhe, Simulation of networks of spiking neurons: A review of tools and strategies, Journal of Computational Neuroscience **23**, 349 (2007).

[20] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang, JAX: composable transformations of Python+NumPy programs (2018), `http://github.com/google/jax`.

[21] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, Array programming with NumPy, Nature **585**, 357 (2020).

[22] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python, Nature Methods **17**, 261 (2020).

[23] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, Pytorch: An imperative style, high-performance deep learning library, in *Advances in Neural Information Processing Systems*, Vol. 32, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (2019) pp. 8024–8035.

[24] I. Babuschkin, K. Baumli, A. Bell, S. Bhupatiraju, J. Bruce, P. Buchlovsky, D. Budden, T. Cai, A. Clark, I. Danihelka, A. Dedieu, C. Fantacci, J. Godwin, C. Jones, R. Hemsley, T. Hennigan, M. Hessel, S. Hou, S. Kapturowski, T. Keck, I. Kemaev, M. King, M. Kunesch, L. Martens, H. Merzic, V. Mikulik, T. Norman, G. Papamakarios, J. Quan, R. Ring, F. Ruiz, A. Sanchez, R. Schneider, E. Sezener, S. Spencer, S. Srinivasan, W. Stokowiec, L. Wang, G. Zhou, and F. Viola, The DeepMind JAX Ecosystem (2020), `http://github.com/deepmind`.

[25] P. Moritz, R. Nishihara, S. Wang, A. Tumanov, R. Liaw, E. Liang, M. Elibol, Z. Yang, W. Paul, M. I. Jordan, and I. Stoica, Ray: A distributed framework for emerging AI applications, in *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)* (USENIX Association, Carlsbad, CA, 2018) pp. 561–577.

[26] J. D. Hunter, Matplotlib: A 2d graphics environment, Computing in Science & Engineering **9**, 90 (2007).

[27] M. A. Petroff, Accessible color sequences for data visualization (2021), arXiv:2107.02270 [cs.GR].

[28] C. Klos, (2024), `https://github.com/chklos/spikegd`.

[29] W. Rudin, *Principles of Mathematical Analysis* (McGraw-Hill, New York, 1976).

[30] M. W. Hirsch and S. Smale, *Differential equations, dynamical systems, and linear algebra*, Pure and applied mathematics No. 60 (Acad. Press, San Diego [u.a.], 1974).

[31] G. Jetschke, *Mathematik der Selbstorganisation* (Harri Deutsch, Frankfurt am Main, 2009).

[32] V. I. Arnold, *Ordinary Differential Equations* (Springer, Berlin, 1992).

[33] H. Heuser, *Lehrbuch der Analysis. Teil 1* (Teubner-Verlag, 1998).

[34] A. Stanojevic, S. Woźniak, G. Bellec, G. Cherubini, A. Pantazi, and W. Gerstner, An exact mapping from relu networks to spiking neural networks, Neural Networks **168**, 74 (2023).

[35] A. Stanojevic, S. Woźniak, G. Bellec, G. Cherubini, A. Pantazi, and W. Gerstner, High-performance deep spiking neural networks with 0.3 spikes per neuron (2023), arXiv:2306.08744 [cs.NE].

[36] T. Hoefler, D. Alistarh, T. Ben-Nun, N. Dryden, and A. Peste, Sparsity in deep learning: pruning and growth for efficient inference and training in neural networks, J. Mach. Learn. Res. **22** (2021).

[37] S. I. Mirzadeh, K. Alizadeh-Vahid, S. Mehta, C. C. del Mundo, O. Tuzel, G. Samei, M. Rastegari, and M. Farajtabar, ReLU strikes back: Exploiting activation sparsity in large language models, in *The Twelfth International Conference on Learning Representations* (2024).

[38] E. O. Neftci, H. Mostafa, and F. Zenke, Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks, IEEE Signal Processing Magazine **36**, 51 (2019).

[39] D. V. Christensen, R. Dittmann, B. Linares-Barranco, A. Sebastian, M. L. Gallo, A. Redaelli, S. Slesazeck, T. Mikolajick, S. Spiga, S. Menzel, I. Valov, G. Milano, C. Ricciardi, S.-J. Liang, F. Miao, M. Lanza, T. J. Quill, S. T. Keene, A. Salleo, J. Grollier, D. Marković, A. Mizrahi, P. Yao, J. J. Yang, G. Indiveri, J. P. Strachan, S. Datta, E. Vianello, A. Valentian, J. Feldmann, X. Li, W. H. P. Pernice, H. Bhaskaran, S. Furber, E. Neftci, F. Scherr, W. Maass, S. Ramaswamy, J. Tapson, P. Panda, Y. Kim, G. Tanaka, S. Thorpe, C. Bartolozzi, T. A. Cleland, C. Posch, S. Liu, G. Panuccio, M. Mahmud, A. N. Mazumder, M. Hosseini, T. Mohsenin, E. Donati, S. Tolu, R. Galeazzi, M. E. Christensen, S. Holm, D. Ielmini, and N. Pryds, 2022 roadmap on neuromorphic computing and engineering, Neuromorphic Computing and Engineering **2**, 022501 (2022).

[40] E. Nordlie, M.-O. Gewaltig, and H. E. Plesser, Towards reproducible descriptions of neuronal network models, PLOS Computational Biology **5**, 1 (2009).

[41] J. Zhuang, T. Tang, Y. Ding, S. Tatikonda, N. Dvornek, X. Papademetris, and J. Duncan, Adabelief optimizer: Adapting stepsizes by the belief in observed gradients, in *NeurIPS 2020 Workshop: Deep Learning through Information Geometry* (2020).

[42] J. Göltz, L. Kriener, A. Baumbach, S. Billaudelle, O. Breitwieser, B. Cramer, D. Dold, A. F. Kungl, W. Senn, J. Schemmel, K. Meier, and M. A. Petrovici, Fast and energy-efficient neuromorphic deep learning with first-spike times, Nature Machine Intelligence **3**, 823 (2021).