A NEURONAL LEAST-ACTION PRINCIPLE FOR REAL-TIME LEARNING IN CORTICAL CIRCUITS

Walter Senn[*,a] **Dominik Dold**[*,a,b,c] **Akos F. Kungl**[a,b] **Benjamin Ellenberger**[a,f] **Jakob Jordan**[a]

Yoshua Bengio[d] **João Sacramento**[e] **Mihai A. Petrovici**[*,a,b]

[*] *Equal contribution*

[a] *Department of Physiology, University of Bern*, [b] *Kirchhoff-Institute for Physics, Heidelberg University*, [c] *European Space Research and Technology Centre, European Space Agency*, [d] *MILA, University of Montreal*, [e] *Department of Computer Science, ETH Zurich*, [f] *Insel Data Science Center, University Hospital Bern.*

ABSTRACT

One of the most fundamental laws of physics is the principle of least action. Motivated by its predictive power, we introduce a neural least-action principle that we apply to motor control. The central notion is the somato-dendritic mismatch error within individual neurons. The principle postulates that the somato-dendritic mismatch errors across all neurons in a cortical network are minimized by the voltage dynamics. Ongoing synaptic plasticity reduces the somato-dendritic mismatch error within each neuron and performs gradient descent on the output cost in real time. The neuronal activity is prospective, ensuring that dendritic errors deep in the network are prospectively corrected to eventually reduce motor errors. The neuron-specific errors are represented in the apical dendrites of pyramidal neurons, and are extracted by a cortical microcircuit that 'explains away' the feedback from the periphery. The principle offers a general theoretical framework to functionally describe real-time neuronal and synaptic processing.

Keywords Least action · neuronal dynamics · synaptic plasticity · error backpropagation · real-time learning

Introduction

Wigner's remark about the 'unreasonable effectiveness' of mathematics in allowing us to understand physical phenomena (Wigner, 1959) is famously contrasted by Gelfand's quip about its 'unreasonable ineffectiveness' in doing the same for biology (Borovik, 2021). Considering the component of randomness that is inherent to evolution, this may not be all that surprising. However, while this argument holds just as well for the brain at the cellular level, ultimately brains are computing devices. At the level of computation, machine learning and neuroscience have revealed near-optimal strategies for information processing and storage, and evolution is likely to have found similar principles through trial and error (Hassabis *et al.,* 2017). Thus, we have reason to hope for the existence of fundamental principles of cortical computation that are similar to those we have found in the physical sciences. Eventually, it is important for such approaches to relate these principles back to brain phenomenology and connect function to structure and dynamics.

In physics, a fundamental measure of 'effort' is the action of a system, which nature seeks to 'minimize'. Given an appropriate description of interactions between the systems constituents, the least-action principle can be used to derive the equations of motion of any physical system (Feynman *et al.,* 2011; Coopersmith, 2017). Here, we suggest that in biological information processing, a similar principle holds for prediction errors, which are of obvious relevance for cognition and behavior. Based on such errors, we formulate a neuronal least-action (NLA) principle which can be used to derive neuronal dynamics and map them to observed dendritic morphologies and cortical microcircuits. Within this framework, local synaptic plasticity at basal and apical dendrites can be derived by stochastic gradient descent on errors. The errors that are minimized refer to the errors in output neurons that are typically thought to represent motor trajectories planed and encoded in cortical motor areas and ultimately in the spinal cord and muscles.

The theory considers the minimization of a mismatch between the effective output trajectory and an internally or externally imposed target trajectory. The strength by which the output trajectory is pushed towards the target trajectory is referred to as 'nudging strength'. For weak nudging, the feedback slightly adapts the internal network states so that the output activities closer follow their targets. For a target that is fixed in time, the dynamics reduces to the Equilibrium Propagation (Scellier & Bengio, 2017). For strong nudging, the output neurons are clamped to tightly follow the target trajectory (Song *et al.,* 2022; Meulemans, Zucchet, *et al.,* 2022). Each network neuron performs gradient descent on its own dendritic prediction error (Urbanczik & Senn, 2014), and with this eventually also contributes to a reduction of the output errors. For both versions, weak and strong nudging, synaptic plasticity in an interneuron feedback circuitry supports the extraction of neuron-specific dendritic errors (Sacramento *et al.,* 2018), providing a cortical implementation of the suggested 'real-time dendritic error propagation (rt-DeEP)'.

The NLA principle can be put into the history of energy-based models used to understand neuronal processing and learning in recurrent neural networks (Hopfield, 1982), artificial intelligence (LeCun *et al.,* 2006), and biological versions of deep learning (Scellier & Bengio, 2017; Richards *et al.,* 2019; Song *et al.,* 2022). As a continuous-time theory of cortical processing that instantaneously corrects neuronal activities throughout the network, it makes a link to optimal feedback control (Todorov & Jordan, 2002) that has recently been considered in terms of energy-based models at equilibria (Meulemans, Zucchet, *et al.,* 2022). It can further be seen in the tradition of predictive coding (Rao & Ballard, 1999; Bastos *et al.,* 2012), where cortical feedback connections try to explain away lower-level activities. Yet, our prospective coding extrapolates from current quantities to predict activity in the future, not the activity at the current point in time, as in classical predictive coding. The NLA principle combines energy-based models with prospective coding in which neuronal integration delays are compensated on the fly (as also done in Haider *et al.,* 2021). The prospective coding within each neuron may overcome putative delays that extend across the whole cortex and would allow for a real-time processing of stimuli while instantaneously correcting for ongoing errors. In this sense, the NLA represents an alternative to the recently suggested forward-forward algorithm which circumvents the delay problem by avoiding error propagation altogether (Hinton, 2022).

The paper is organized as follows: we first define the prospective somato-dendritic mismatch error, construct out of this the mismatch energy of a network, and 'minimise' this energy to obtain the error-corrected, prospective voltage dynamics of the network neurons. We then show that the prospective error coding leads to an instantaneous and joint processing of low-pass filtered input signals and backpropagated errors. Applied to motor control, the instantaneous processing is interpreted as a moving equilibrium hypothesis according to which sensory inputs, network state, motor commands and muscle feedback are in a self-consistent equilibrium at any point of the movement. We then derive a local learning rule that globally minimizes the somato-dendritic mismatch errors across the network, and show how this learning can be implemented through error-extracting cortical microcircuits and dendritic predictive plasticity.

Results

Somato-dendritic mismatch errors and the Lagrangian of cortical circuits

We consider a network of neurons – identified as pyramidal cells – with firing rates $r_i(t)$ in continuous time t. The somatic voltage u_i of pyramidal neuron i is driven by the close-by basal input current, $\sum_j W_{ij} r_j$, with presynaptic rates r_j and synaptic weights W_{ij}, and an additional distal apical input e_i that will be learned to represent a prospective prediction error at any moment in time (Fig. 1a). While in classical rate-based neuron models the firing rate r_i of a neuron is a function of the somatic voltage, $\rho(u_i)$, we postulate that the firing rate is prospective and extrapolates from $\rho(u_i)$ into the future with the temporal derivative, $r_i = \rho(u_i) + \tau\dot{\rho}(u_i)$, with $\dot{\rho}(u_i)$ representing the temporal derivative of $\rho(u_i(t))$. There is experimental evidence for such prospective coding in cortical pyramidal neurons to which we return later (Fig. 2a).

The second central notion of the theory is the prospective somato-dendritic mismatch error in the individual network neurons, $e_i(t)$. It is defined as a mismatch between the prospective voltage, $u_i + \tau\dot{u}_i$, and the weighted prospective input rates, $e_i = (u_i + \tau\dot{u}_i) - \sum_j W_{ij} r_j$. If the prospective error is low-pass filtered with time constant τ, it takes the form $\bar{e}_i = u_i - \sum_j W_{ij}\bar{r}_j$, where $\bar{r}_j$ is the corresponding low-pass filtered firing rate of the presynaptic neuron j (Methods). We refer to $\bar{e}_i$ as somato-dendritic mismatch error of neuron i.

We next interpret the mismatch error $\bar{e}_i$ in terms of the morphology of pyramidal neurons with basal and apical dendrites. While the error is formed in the apical dendrite, this error is added to the somatic voltage and, from the perspective of the basal inputs, it becomes a somato-dendritic mismatch error. The mismatch error tells the difference between 'what a neuron does', which is based on the somatic voltage u_i, and 'what the basal inputs think it should do', which is based on its own input $\sum W_{ij}\bar{r}_j$ (Fig. 1a2). The two quantities may deviate because the neuron gets the apical input $\bar{e}_i$ that integrates in the soma, $u_i = \sum_j W_{ij}\bar{r}_j + \bar{e}_i$, but not in the basal dendrite. What cannot be predicted from the somatic

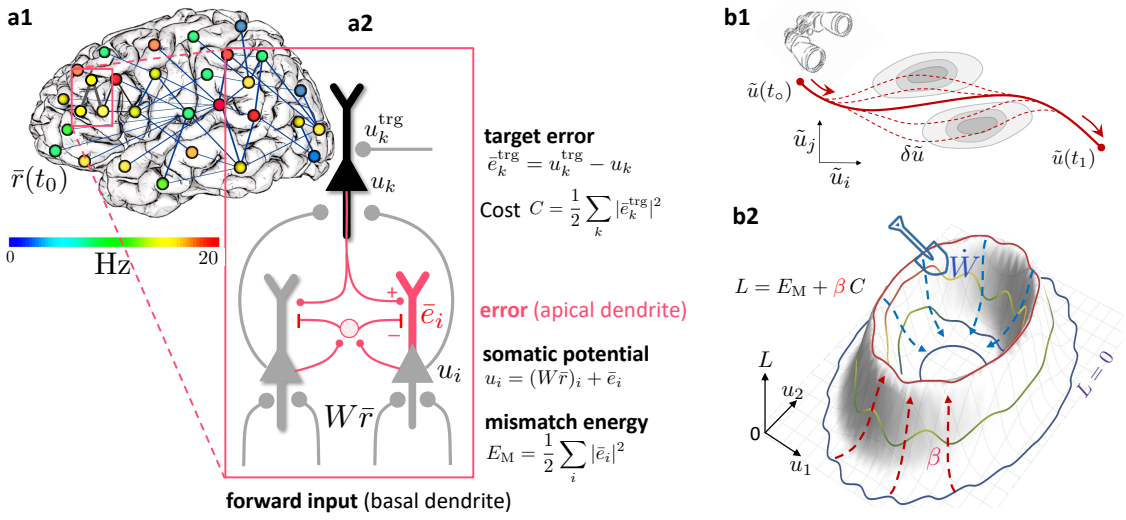

Figure 1: **Somato-dendritic mismatch energies and the neuronal least-action (NLA) principle. (a1)** Sketch of a cross-cortical network of pyramidal neurons described by NLA. **(a2)** Correspondence between elements of NLA and biological observables such as membrane voltages and synaptic weights. **(b1)** The NLA principle postulates that small variations $\delta\tilde{\boldsymbol{u}}$ (dashed) of the trajectories $\tilde{\boldsymbol{u}}$ (solid) leave the action invariant, $\delta A = 0$. It is formulated in the look-ahead coordinates $\tilde{\boldsymbol{u}}$ (symbolized by the spyglass) in which 'hills' of the Lagrangian (shaded grey zones) are foreseen by the prospective voltage so that the trajectory can turn by early enough to surround them. **(b2)** In the absence of output nudging ($\beta = 0$), the trajectory $\boldsymbol{u}(t)$ is solely driven by the sensory input, and prediction errors and energies vanish ($L = 0$, outer blue trajectory at bottom). When nudging the output neurons towards a target voltage ($\beta > 0$), somato-dendritic prediction errors appear, the energy increases (red dashed arrows symbolising the growing 'volcano') and the trajectory $\boldsymbol{u}(t)$ moves out of the $L = 0$ hyperplane, riding on top of the 'volcano' (red trajectory). Synaptic plasticity $\dot{W}$ reduces the somato-dendritic mismatch along the trajectory by optimally 'shoveling down the volcano' (blue dashed arrows) while the trajectory settles in a new place on the $L = 0$ hyperplane (inner blue trajectory at bottom).

voltage u_i by the basal input remains as 'somato-basal' mismatch error, $u_i - \sum_j W_{ij}\bar{r}_j$, and this is just the apical error $\bar{e}_i$.

Associated with this mismatch error is the somato-dendritic mismatch energy defined for each network neuron $i \in \mathcal{N}$ as the squared mismatch error,

$$E_i^{\mathrm{M}} = \tfrac{1}{2}\bar{e}_i^2 = \tfrac{1}{2}\left(u_i - \textstyle\sum_j W_{ij}\bar{r}_j\right)^2 . \tag{1}$$

On a subset of output neurons of the whole network, $\mathcal{O} \subseteq \mathcal{N}$, a cost is defined as a function of the somatic voltage and some instructive reference signal such as a targets or a reward. When a target trajectory $u_o^*(t)$ is available, the cost is defined at each time point as a squared target error

$$C_o = \tfrac{1}{2}\bar{e}_o^* = \tfrac{1}{2}\left(u_o^* - u_o\right)^2 . \tag{2}$$

Much more general cost functions and mismatch energies are conceivable, encompassing e.g. conductance-based neurons or including further dynamic variables (see SI). The cost represents a performance measure for the entire network that produces the output voltages $u_o(t)$ in response to some input rates $\boldsymbol{r}_{\mathrm{in}}(t)$. The cost directly relates to behavioral or cognitive measures such as the ability of an animal or human to perform a particular task in real time. The target could be provided by explicit external supervision, for example target movements in time encoded by $u_o^*(t)$, it could represent an expected reward signal, or it could arise via self-supervision from other internal prediction errors.

We define the Lagrangian (or total 'energy') of the network as a sum across all mismatch energies and costs, weighted by the nudging strength β of the output neurons,

$$L = \sum_{i\in\mathcal{N}} E_i^{\mathrm{M}} + \beta \sum_{o\in\mathcal{O}} C_o = \frac{1}{2}\sum_{i\in\mathcal{N}}\left(u_i - \textstyle\sum_j W_{ij}\bar{r}_j\right)^2 + \frac{\beta}{2}\sum_{o\in\mathcal{O}}\left(u_o^* - u_o\right)^2 . \tag{3}$$

The low-pass filtered presynaptic rates, $\bar{r}_j$, also encompass the external input neurons. Due to the prospective coding, the Lagrangian can be minimal at any moment in time while the network dynamics evolves. This is different from

the classical predictive coding (Rao & Ballard, 1999) and energy-based approaches (Scellier & Bengio, 2017; Song *et al.,* 2022), where a stimulus needs to be fixed in time while the network relaxes to a steady state, and only there the prediction error is minimized. Since we postulated that the firing rates are prospective and linearly extrapolate into the future, $r_i = \rho(u_i) + \tau\dot{\rho}(u_i)$, and because any quantity can be reconstructed from its low-pass filtering at any point in time, $r_i = \bar{r}_i + \tau\dot{\bar{r}}_i$, we conclude that the low-pass filtered rate is a function of the instantaneous somatic voltage, $\bar{r}(t) = \rho(u(t))$, see SI. As a consequence, also the Lagrangian becomes a function of the instantaneous voltage $u(t)$ and does not depend on past voltage values, although it depends on past firing rates.

The previous mathematical operation of extrapolating from the voltage to future rates, and going back to the current voltage via low-pass filtering, is the key ingredient for the magically sounding instantaneous processing, see Fig. 2c.

To get the instantaneity it is not really necessary to precisely look into the future. It is only necessary to undo the temporal operation of the postsynaptic neuron, and this is achieved by encoding the presynaptic voltage in the prospective rate. We come back to this instantaneity, including its practical limitations, in the overnext section.

The least-action principle expressed for prospective voltages

Motivated by the prospective firing in pyramidal neurons, we postulate that cortical networks strive to look into the future to prevent instantaneous errors. Each neuron tries to move along a trajectory that minimizes its own mismatch error $\bar{e}_i$ across time (Fig. 1b1). The 'neuronal currency' with which each neuron 'trades' with others to choose its own error-minimizing trajectory is the future discounted membrane potential,

$$\tilde{u}(t) = \frac{1}{\tau}\int_t^\infty u(t')\boldsymbol{e}^{(t-t')/\tau}\mathrm{d}t' \ . \tag{4}$$

These prospective voltages $\tilde{u}$ are the 'canonical coordinates' entering the NLA principle, and in these prospective coordinates the overall network searches for a 'least action trajectory'. Since from $\tilde{u}$ we can recover the instantaneous voltage via $u = \tilde{u} - \tau\dot{\tilde{u}}$ (see SI), we can replace u in the Lagrangian and obtain L as a function of our new canonical coordinates $\tilde{u}$ and the 'velocities' $\dot{\tilde{u}}$, i.e. $L = L\left[\tilde{\boldsymbol{u}}, \dot{\tilde{\boldsymbol{u}}}\right]$, where bold fonts represent vectors. Inspired by the least-action principle from physics, we can now define the neuronal action A as a time integral of the Lagrangian,

$$A = \int_{t_1}^{t_2} L\left[\tilde{\boldsymbol{u}}(t), \dot{\tilde{\boldsymbol{u}}}(t)\right]\mathrm{d}t \ . \tag{5}$$

The NLA principle postulates that the trajectory $\tilde{\boldsymbol{u}}(t)$ keeps the action A stationary with respect to small variations $\delta\tilde{\boldsymbol{u}}$ (Fig. 1b1). In other words, nature chooses a trajectory such that, when deviating a little bit from it, say by $\delta\tilde{\boldsymbol{u}}$, the value of A will not change (or at most up to second order in the variation), formally $\delta A = 0$. The equations of motion that keep the action stationary are known to satisfy the Euler-Lagrange equations

$$\frac{\partial L}{\partial \tilde{u}_i} - \frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{\tilde{u}}_i} = 0 \ . \tag{6}$$

Applying these equations to our Lagrangian yields a prospective version of the classical leaky integrator voltage dynamics, with rates $\boldsymbol{r}$ and errors $\boldsymbol{e}$ that are looking into the future (Methods),

$$\tau\dot{\boldsymbol{u}} = -\boldsymbol{u} + \boldsymbol{W}\boldsymbol{r} + \boldsymbol{e} \ , \tag{7a}$$

$$\bar{\boldsymbol{e}} = \bar{\boldsymbol{r}}'_{\text{net}} \cdot \boldsymbol{W}_{\text{net}}^{\mathrm{T}}\bar{\boldsymbol{e}} + \beta\bar{\boldsymbol{e}}^* \ . \tag{7b}$$

The '$\cdot$' denotes the component-wise product, and the weight matrix splits into weights from input neurons and weights from network neurons, $\boldsymbol{W} = (\boldsymbol{W}_{\text{in}}, \boldsymbol{W}_{\text{net}})$. While for output neurons a target error can be defined, $\bar{e}_o^* = u_o^* - u_o$, for non-output neurons i no target exist and we hence set $\bar{e}_i^* = 0$. In a control theoretic framework, the neuronal dynamics (Eq. 7a) represents the state trajectory, and the adjoint error dynamics (Eq. 7b) represents the integrated costate trajectory (Todorov, 2006).

Formally, the voltage dynamics in Eq. 7a specifies an implicit differential equation since $\dot{\boldsymbol{u}}(t)$ also appears on the right-hand side. This is because the prospective rates $\boldsymbol{r} = \rho(\boldsymbol{u}) + \tau\dot{\rho}(\boldsymbol{u})$ imply $\dot{\boldsymbol{u}}$ through $\dot{\rho}(\boldsymbol{u}) = \rho'(\boldsymbol{u})\cdot\dot{\boldsymbol{u}}$. Likewise, the prospective errors $\boldsymbol{e} = \bar{\boldsymbol{e}} + \tau\dot{\bar{\boldsymbol{e}}}$, with $\bar{\boldsymbol{e}}$ given in Eq. 7b and plugged into Eq. 7a, imply $\dot{\boldsymbol{u}}$ through $\dot{\bar{\boldsymbol{e}}}(\boldsymbol{u}) = \bar{\boldsymbol{e}}'(\boldsymbol{u})\cdot\dot{\boldsymbol{u}}$. Nevertheless, the voltage dynamics can be run by replacing $\dot{\boldsymbol{u}}(t)$ on the right-hand side of Eq. 7a by the temporal derivative $\dot{\boldsymbol{u}}(t - dt)$ from the previous time step (technically, the Hessian $(\boldsymbol{1} - \boldsymbol{W}\boldsymbol{\rho}' - \bar{\boldsymbol{e}}')$ is required to be strictly positive definite, see Methods and SI). This ensures that the voltage dynamics of Eq. 7 can be implemented in cortical neurons with a prospective firing and a prospective dendritic error (see Fig. 2).

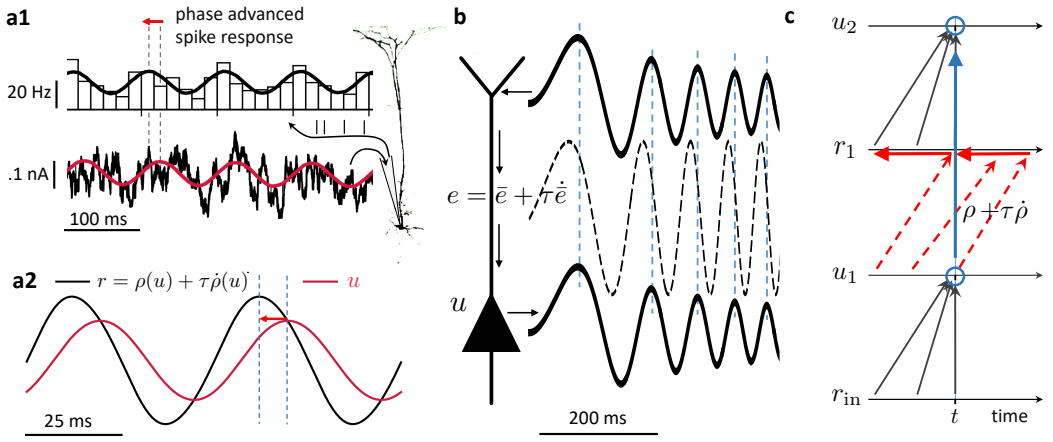

Figure 2: **Prospective coding in cortical pyramidal neurons enables instantaneous voltage-to-voltage transfer.** **(a1)** The instantaneous spike rate of cortical pyramidal neurons (top) in response to sinusoidally modulated noisy input current (bottom) is phase-advanced with respect to the input (adapted from Köndgen *et al.,* 2008). **(a2)** Similiarly, in NLA, the instantaneous firing rate of a model neuron ($r = \rho(u) + \tau\dot{\rho}(u)$, black) is phase-advanced with respect to the underlying voltage (u, red, postulating that the low-pass filtered rate is a function of the voltage, $\bar{r} = \rho(u)$). **(b)** Dendritic input in the apical tree (here called $\bar{e}$) is instantaneously causing a somatic voltage modulation (u, (modeling data from Ulrich, 2002)). The low-pass filtering with τ along the dendritic shaft is compensated by a lookahead mechanism in the dendrite ($e = \bar{e} + \tau\dot{\bar{e}}$). In Ulrich (2002) a phase advance is observed even with respect to the dendritic input current, not only the dendritic voltage, although only for slow modulations (as here). **(c)** While the voltage of the first neuron (u_1) integrates the input rates r_{in} from the past (bottom black upward arrows), the output rate r_1 of that first neuron looks ahead in time, $r_1 = \rho(u_1) + \tau\dot{\rho}(u_1)$ (red dashed arrows pointing into the future). The voltage of the second neuron (u_2) integrates the prospective rates r_1 (top black upwards arrows). By doing so, it inverts the lookahead operation, resulting in an instantaneous transfer from $u_1(t)$ to $u_2(t)$ (blue arrow and circles).

The error expression in Eq. 7b is reminiscent of error backpropagation (Rumelhart *et al.,* 1986) and can in fact be mathematically linked to this (Methods). Formally, the errors are backpropagated via transposed network matrix, $\boldsymbol{W}_{\text{net}}^{\text{T}}$, modulated by $\bar{r}_i'$, the derivative of $\bar{r}_i = \rho(u_i)$ with respect to the underlying voltage. While the transpose can be constructed with various local methods (see Akrout *et al.,* 2019; Max *et al.,* 2022) in our simulations we mainly adhere to the phenomenon of feedback alignment (Lillicrap, Cownden, *et al.,* 2016) and consider fixed and randomized feedback weights $\boldsymbol{B}$ (unless stated differently). Recent control theoretical work is exploiting the same prospective coding technique as expressed in Eq. 7 to tackle general time-varying optimization problems (see Simonetto *et al.,* 2020 for a review and the SI for the detailed connection).

Prospective coding in neurons and instantaneous propagation

The prospective rates and errors entering via $\boldsymbol{r}$ and $\boldsymbol{e}$ in the NLA (Eq. 7) are consistent with the prospective coding observed in cortical pyramidal neurons *in vitro* (Köndgen *et al.,* 2008). Upon sinusoidal current injection into the soma, the somatic firing rate is advanced with respect to its voltage (Fig. 2a), effectively compensating for the delay caused by the current integration. Likewise, sinusoidal current injection in the apical tree causes a lag-less voltage response in the soma (Fig. 2b, Ulrich, 2002). While the rates and errors in general can be reconstructed from their low-pass filterings via $\boldsymbol{r} = \bar{\boldsymbol{r}} + \tau\dot{\bar{\boldsymbol{r}}}$ and $\boldsymbol{e} = \bar{\boldsymbol{e}} + \tau\dot{\bar{\boldsymbol{e}}}$, they become prospective in time because $\bar{\boldsymbol{r}}$ and $\bar{\boldsymbol{e}}$ are themselves instantaneous functions of the voltage $\boldsymbol{u}$, and hence $\boldsymbol{r}$ and $\boldsymbol{e}$ depend on $\dot{\boldsymbol{u}}$. The derivative of the membrane potential implicitly also appears in the firing mechanism of Hodgkin-Huxley-type conductances, with a quick depolarization leading to a stronger sodium influx due to the dynamics of the gating variables (Hodgkin & Huxley, 1952). This advances the action potential as compared to a firing that would only depend on $\boldsymbol{u}$, not $\dot{\boldsymbol{u}}$, giving an intuition of how such a prospective coding may arise. A similar prospective coding has been observed for retinal ganglion cells (Palmer *et al.,* 2015) and cerebellar Purkinje cells (Ostojic *et al.,* 2015), making a link from the visual input to the motor control.

To understand the instantaneous propagation through the network, we low-pass filter the dynamic equation $\boldsymbol{u} + \tau\dot{\boldsymbol{u}} = \boldsymbol{W}\boldsymbol{r} + \boldsymbol{e}$ (obtained by rearranging Eq. 7a), with $\bar{\boldsymbol{e}}$ given by Eq. 7b, to obtain the somatic voltage $\boldsymbol{u} = \boldsymbol{W}\,\bar{\boldsymbol{r}}(\boldsymbol{u}) + \bar{\boldsymbol{e}}(\boldsymbol{u})$. At any point in time, the voltage is in a moving equilibrium between forward and backpropagating inputs. Independently of the network architecture, whether recurrent or not, the output is an instantaneous function of the low-pass filtered input and a putative correction towards the target, $\boldsymbol{u}_{\boldsymbol{o}}(t) = \boldsymbol{F}_W(\bar{\boldsymbol{r}}_{\text{in}}(t), \bar{\boldsymbol{e}}_{\boldsymbol{o}}^*(t))$, see Fig. 2c and Methods. The mapping

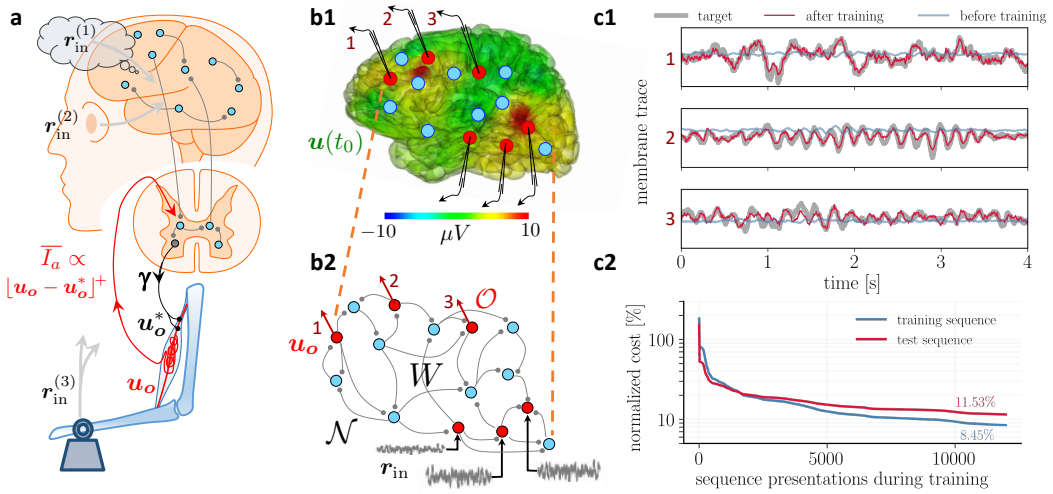

Figure 3: **Moving equilibrium hypothesis for motor control and real-time learning of cortical activity. (a)** A voluntary movement trajectory can be specified by the target length of the muscles in time, $\boldsymbol{u}_{\boldsymbol{o}}^*$, encoded through the γ-innervation of muscle spindles, and the deviation of the effective muscle lengths from the target, $\boldsymbol{u}_{\boldsymbol{o}} - \boldsymbol{u}_{\boldsymbol{o}}^* = -\bar{\boldsymbol{e}}_{\boldsymbol{o}}^*$. The I_a-afferents emerging from the spindles prospectively encode the error, so that their low-pass filtering is roughly proportional to the length deviation, truncated at zero (red). The moving equilibrium hypothesis states that low-pass filtered inputs encoding the movement plan, $\bar{\boldsymbol{r}}_{\text{in}}^{(1)}$, the sensory inputs, $\bar{\boldsymbol{r}}_{\text{in}}^{(2)}$, and the condition of the plant, $\bar{\boldsymbol{r}}_{\text{in}}^{(3)}$, instantaneously generate the muscle lengths, $\boldsymbol{u}_{\boldsymbol{o}} = \boldsymbol{F}_W(\bar{\boldsymbol{r}}_{\text{in}}, \bar{\boldsymbol{e}}_{\boldsymbol{o}}^*)$, and are at any point in time in an instantaneous equilibrium with the error feedback from the spindles (such that Eq. 7 is satisfied). **(b1)** Intracortical iEEG activity recorded from 56 deep electrodes and projected to the brain surface. Red nodes symbolize the 56 iEEG recording sites modeled alternately as input or output neurons, and blue nodes symbolize the 40 'hidden' neurons for which no data is available, but used to reproduce the iEEG activity. **(b2)** Corresponding NLA network. During training, the voltages of the output neurons were nudged by the iEEG targets (black input arrows, but for all red output neurons). During testing, nudging was removed for 14 out of these 56 neurons (here, represented by neurons 1, 2, 3). **(c1)** Voltage traces for the 3 example neurons in a2, before (blue) and after (red) training, overlaid with their iEEG target traces (grey). **(c2)** Total cost, integrated over a window of 8 s of the 56 output nodes during training with sequences of the same duration. The cost for the test sequences was evaluated on a 8 s window not used during training.

again expresses an instantaneous propagation of voltages throughout the network in response to both, the low-pass filtered input $\bar{\boldsymbol{r}}_{\text{in}}$ and feedback error $\bar{\boldsymbol{e}}_{\boldsymbol{o}}^*$. This instantaneity is independent of the network size, and in a feed-forward network is independent of its depths (see also Haider *et al.,* 2021, where the instantaneity is on the rates, not the voltages). In the absence of the look-ahead activity, each additional layer slows down the network relaxation time.

Notice that an algorithmic implementation of the time-continuous dynamics of a N-layer feedforward network would still need N calculation steps until information from layer 1 reaches layer N. However, this does not imply that an analog implementation of the prospective dynamics will encounter delays. To see why, consider a finite step-change $\Delta\boldsymbol{u}_1$ in the voltage of layer 1. In the absence of the look-ahead, $\Delta\boldsymbol{u}_1$ were mapped within the infinitesimal time interval dt to an infinitesimal change $d\boldsymbol{u}_2$ in the voltages of layer 2. But with a prospective firing rate, $\boldsymbol{r}_1 = \rho(\boldsymbol{u}_1) + \tau\rho'(\boldsymbol{u}_1)\,\dot{\boldsymbol{u}}_1$, a step-change $\Delta\boldsymbol{u}_1$ translates to a delta-function in $\boldsymbol{r}_1$, this in turn to a step-change in the low-pass filtered rates $\Delta\bar{\boldsymbol{r}}_1$, and therefore within dt to a step-change $\Delta\boldsymbol{u}_2$ in the voltages $\boldsymbol{u}_2$ of the postsynaptic neurons (Fig. 2c). Iterating this argument, a step-change $\Delta\boldsymbol{u}_1$ propagates 'instantaneously' through N layers within the 'infinitesimal' time interval $N\,dt$ to a step-change $\Delta\boldsymbol{u}_N$ in the last layer. When run in a biophysical device in continuous time that exactly implements the dynamical equations (Eq. 7), the implementation becomes an instantaneous computation. Yet, in a biophysical device information has to be moved across space. This typically introduces further propagation delays that may not be captured in our formalism where low-pass filtering and prospective coding cancel each other exactly. Nevertheless, analog computation in continuous time, as formalized here, offers an idea to 'instantaneously' realize an otherwise time consuming numerical recipe run on time-discrete computing systems that operate with a finite clock cycle.

Prospective control and the moving equilibrium hypothesis

Crucially, at the level of the voltage dynamics (Eq. 7a) the correction is based on the prospective error $\boldsymbol{e}$. This links our framework to optimal control theory and motor control where delays are also taken into account, so that a movement can be corrected early enough (Wolpert & Ghahramani, 2000; Todorov & Jordan, 2002; Todorov, 2004). The link between energy-based models and optimal control was recently drawn for strong nudging ($\beta \to \infty$) to learn individual equilibrium states (Meulemans, Zucchet, *et al.,* 2022). Our prospective error $\boldsymbol{e}(t)$ appears as a 'controller' that, when looking at the output neurons, pushes the voltage trajectories towards the target trajectories. Depending on the nudging strength β, the control is tighter or weaker. For infinitely large β, the voltages of the output neurons are clamped to the time-dependent target voltages, $u_o = u_o^*$ (implying $e_o^* = 0$), while their errors, $\bar{e}_o = u_o - (\boldsymbol{W}\bar{\boldsymbol{r}})_o$, instantaneously correct all network neurons. For small β, the output voltages are only weakly controlled, and they are dominated by the forward input, $u_o \approx (W\bar{\boldsymbol{r}})_o$.

In the context of motor control, our network mapping $\boldsymbol{u_o} = \boldsymbol{F}_W(\bar{\boldsymbol{r}}_{\text{in}}, \bar{\boldsymbol{e}}_{\boldsymbol{o}}^*)$ can be seen as a forward internal model that quickly calculates an estimate of the future muscle length $\boldsymbol{u_o}$ based on some motor plans, sensory inputs, and the current proprioceptive feedback (Fig. 3a). Forward models help to overcome delays in the execution of the motor plan by predicting the outcome, so that the intended motor plans and commands can be corrected on the fly (Kawato, 1999; Wolpert & Ghahramani, 2000).

The observation that muscle spindles prospectively encode the muscle length and velocity (Dimitriou & Edin, 2010)) suggests that the prospective coding in the internal forward model mirrors the prospective coding in the effective forward pathway. This forward pathway leads from motor plan to spindle feedback, integrating also cerebellar and brainstem feedback (Kawato, 1999). Based on the motor plans, the intended spindle lengths and the effective muscle innervation are communicated via descending pathway to activate the γ- and α-motoneurons, respectively (Li *et al.,* 2015). The transform from the intended arm trajectory to the intended spindle lengths via γ-innervation is mainly determined by the joint geometry. The transform from the intended arm trajectory to the force generating α-innervation, however, needs to also take account of the internal and external forces, and this is engaging our network $\boldsymbol{W}$.

When we prepare an arm movement, spindles in antagonistic muscles pairs that measure the muscle length are tightened or relaxed before the movement starts (Papaioannou & Dimitriou, 2021). According to the classical equilibrium-point hypothesis (Feldman & Levin, 2009; Latash, 2018), top-down input adjusts the activation threshold of the spindles through (γ-)innervation from the spinal cord so that slight deviations from the equilibrium position can be signaled (Fig. 3a). We postulate that this γ-innervation acts also during the movement, setting an instantaneous target $\boldsymbol{u_o^*}(t)$ for the spindle lengths. The effective lengths of the muscle spindles is $\boldsymbol{u_o}$, and the spindles are prospectively signaling back the deviation from the target through the I_a-afferents (Dimitriou & Edin, 2010; Dimitriou, 2022). The low-pass filtered I_a-afferents may be approximated by a threshold-nonlinearity, $\overline{I_a} = \beta\lfloor \boldsymbol{u_o} - \boldsymbol{u_o^*} \rfloor^+$, with β being interpreted as spindle gain (Latash, 2018). Combining the feedback from agonistic and antagonistic muscle pairs allows for extracting the scaled target error $\beta\,\bar{\boldsymbol{e}}_{\boldsymbol{o}}^* = \beta(\boldsymbol{u_o^*} - \boldsymbol{u_o})$. Taking account of the prospective feedback, we postulate the *moving equilibrium hypothesis* according to which the instructional inputs, $\bar{\boldsymbol{r}}_{\text{in}}$, the spindle feedback, $\beta\,\bar{\boldsymbol{e}}_{\boldsymbol{o}}^*$, and the muscle lengths, $\boldsymbol{u_o}$, are at any point of the movement in a dynamic equilibrium. The moving equilibrium hypothesis extends the classical equilibrium-point hypothesis from the spatial to the temporal domain.

With regard to the interpretation of the prospective feedback error $\boldsymbol{e_o^*}$ as spindle activity, it is worth noticing that in humans the spindle activity is not only ahead of the muscle activation (Dimitriou & Edin, 2010), but also shares the property of a motor error (Dimitriou, 2016). The experiments show that during the learning of a gated hand movement, spindle activity is initially stronger when making movement errors, and it returns back to baseline with the success of learning. We next address how the synaptic strengths $\boldsymbol{W}$ specifying the forward model can be optimally adapted to capture this learning.

Local plasticity at basal synapses minimizes the global cost in real time

The general learning paradigm starts with input time series $r_{\text{in}(t),i}$ and target time series $u_o^*(t)$, while assuming that the target series are an instantaneous function of the low-pass filtered input series, $\boldsymbol{u_o^*}(t) = \boldsymbol{F}^*(\bar{\boldsymbol{r}}_{\text{in}}^*(t))$. The low-pass filtering in the individual inputs could be with respect to any time constant $\tau_{\text{in},i}^*$ (that may also be learned, see SI), but for the network neurons we assume the same time constants for the voltage integration and the prospective rate. The goal of learning is to adapt the synaptic strengths $\boldsymbol{W}$ in the student network so that this approximates the target mapping, $\boldsymbol{F}_W \approx \boldsymbol{F}^*$. This will also reduce the cost C defined on the output neurons in terms of the deviation of the voltage from the target, $u_o^* - u_o$ (Eq. 2).

The problem of changing synaptic weights to correct the behavior of downstream neurons, potentially multiple synapses away, is typically referred to as the credit assignment problem and is notoriously challenging in physical or biological

substrates operating in continuous time. A core aspect of the NLA principle is how it relates the cost C to the Lagrangian L and eventually to somato-dendritic prediction errors $\bar{\boldsymbol{e}}$ that can be reduced through synaptic plasticity $\dot{\boldsymbol{W}}$. We define this synaptic plasticity as partial derivative of the Lagrangian with respect to the weights, $\dot{\boldsymbol{W}} \propto -\frac{\partial L}{\partial \boldsymbol{W}} = \bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^{\mathrm{T}}$. Since the somato-dendritic mismatch error is $\bar{\boldsymbol{e}} = \boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}}$, this leads to the local learning rule of the form 'postsynaptic error times low-pass filtered presynaptic rate',

$$\dot{\boldsymbol{W}} = \eta\,(\boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}})\,\bar{\boldsymbol{r}}^{\mathrm{T}}\,. \tag{8}$$

This plasticity rule runs simultaneously to the neuronal dynamics in the presence of a given nudging strength β that tells how strongly the voltage of an output neurons is pushed towards the target, $u_o \to u_o^*$. The learning rule is local in space since $\boldsymbol{W}\bar{\boldsymbol{r}}$ is represented as voltage of the basal dendrites, and the somatic voltage $\boldsymbol{u}$ may be read out at the synaptic site on the basal dendrite from the backpropagating action potentials that sample $\boldsymbol{u}$ at a given time (Urbanczik & Senn, 2014). The basal voltage $\boldsymbol{W}\bar{\boldsymbol{r}}$ becomes the dendritic prediction of the somatic activity $\boldsymbol{u}$, interpreting Eq. 8 as 'dendritic predictive plasticity'.

We have derived the neuronal dynamics as a path that keeps the action stationary. Without external teaching signal, the voltage trajectory wriggles on the bottom of the energy landscape ($L = 0$, Fig. 1b1). If the external nudging is turned on, $\beta > 0$, errors emerge and hills grow out of the landscape. The trajectory still tries to locally minimize the action, but it is lifted upwards on the hills ($L > 0$, Fig. 1b2). Synaptic plasticity reshapes the landscape so that, while keeping β fixed, the errors are reduced and the landscape again flattens. The transformed trajectory settles anew in another place (inside the 'volcano' in Fig. 1b2). Formally, the local plasticity rule (Eq. 8) is shown to perform gradient descent on the Lagrangian. In the energy landscape picture, plasticity 'shovels off' earth along the voltage path so that this is lowered most efficiently. The error that is back-propagated through the network tells at any point on the voltage trajectory how much to 'dig' in each direction, i.e. how to adapt the basal input in each neuron in order to optimally lower the local error.

The following theorem tells that synaptic plasticity $\dot{\boldsymbol{W}}$ pushes the network mapping $\boldsymbol{u_o} = \boldsymbol{F}_W(\bar{\boldsymbol{r}}_{\mathrm{in}})$ towards the target mapping $\boldsymbol{u_o^*} = \boldsymbol{F}^*(\bar{\boldsymbol{r}}_{\mathrm{in}})$ at any moment in time. The convergence of the mapping is a consequence of the fact the plasticity reduces the Lagrangian $L = E^{\mathrm{M}} + \beta C$ along its gradient.

Theorem 1 (real-time Dendritic Error Propagation, rt-DeEP) *Consider an arbitrary network $\boldsymbol{W}$ with voltage and error dynamics following Eq. 7. Then the local plasticity rule $\dot{\boldsymbol{W}} \propto \bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^T$ (Eq. 8), acting at each moment along the voltage trajectories, is gradient descent*

(i) on the Lagrangian L for any nudging strength $\beta \geq 0$, i.e. $\bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^T = -\frac{\mathrm{d}L}{\mathrm{d}\boldsymbol{W}}$, with $\lim_{\beta\to\infty} \bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^T = -\frac{\mathrm{d}E^M}{\mathrm{d}\boldsymbol{W}} \propto \dot{\boldsymbol{W}}$.

(ii) on the cost C for small nudging, $\beta \to 0$, while up-scaling the error to $\frac{1}{\beta}\bar{\boldsymbol{e}}$, i.e. $\lim_{\beta\to 0} \frac{1}{\beta}\bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^T = -\frac{\mathrm{d}C}{\mathrm{d}\boldsymbol{W}} \propto \dot{\boldsymbol{W}}$.

The gradient statements hold at any point in time (long enough after initialization), even if the input trajectories $\boldsymbol{r}_{\mathrm{in}}(t)$ contain delta-functions and the target trajectories $\boldsymbol{u_o^*}(t)$ contain step-functions.

Loosely speaking, the NLA enables the network to localize an otherwise global problem: what is good for a single neuron becomes good for the entire network. In the limit of strong nudging ($\beta \to \infty$), the learning rule performs gradient descent on the mismatch energies E_i^{M} in the individual neurons in the network. If the network architecture is powerful enough so that after learning all the mismatch energies vanish, $E_i^{\mathrm{M}} = 0$, then the cost will also vanish, $C = \frac{1}{2}\|\boldsymbol{u_o^*} - \boldsymbol{u_o}\|^2 = 0$. This is because for the output neurons the mismatch error includes the target error. In the limit of weak nudging ($\beta \to 0$), the learning rule performs gradient descent on C, and with this also finds a local minimum of the mismatch energies.

In the case of weak nudging and a single equilibrium state, the NLA algorithm reduces to the Equilibrium Propagation algorithm (Scellier & Bengio, 2017) that minimizes the cost C for a constant input and a constant target. In the case of strong nudging and a single equilibrium state, the NLA principle reduces to the Least Control Principle (Meulemans, Zucchet, *et al.,* 2022) that minimizes the mismatch energy E^{M} for a constant input and a constant target. While in the Least Control Principle the inputs and outputs are clamped to fixed values, the output errors are backpropagated and the network equilibrates in a steady state where the corrected network activities reproduce the clamped output activities. This state is called the 'prospective configuration' in (Song *et al.,* 2022) because neurons deep in the network are informed about the distal target and are correspondingly adapted to be consistent with this distal target. In the NLA principle, after an initial transient, this relaxation process occurs on the fly while inputs and targets dynamically change, and hence the network moves along a continuous sequence of prospective configurations.

In the motor control example, the theorem tells that a given target motor trajectory $\boldsymbol{u_o^*}(t)$ is learned to be re-produced by the forward model $\boldsymbol{u_o}(t) = \boldsymbol{F}_W(\bar{\boldsymbol{r}}_{\mathrm{in}}(t))$ through the dendritic predictive plasticity (Eq. 8), applied in the various

layers of the forward model. We give an example below for such a learning that involves different sensory modalities to train the forward model.

Reproducing intracortical EEG recordings and recognizing handwritten digits. To illustrate the framework, we consider a recurrently connected network that learns to represent intracortical electroencephalogram (iEEG) data from epileptic patients (Fig. 3b). For each electrode, we assign a neuron within this network to represent the activity of the cell cluster recorded in the corresponding iEEG signal via its membrane potential. During learning, a randomly selected subset of electrode neurons are nudged towards the target activity from recorded data while learned to be reproduced by the other neurons. After learning, we can present only a subset of electrode neurons with previously unseen recordings and observe how the activity of the other neurons closely matches the recordings of their respective electrodes (Fig. 3c). The network derived from NLA is thus able to learn complex correlations between signals evolving in real time by encoding them into its connectivity structure.

As an example of visual processing in the NLA framework, we next consider a well-studied image recognition task, here reformulated in a challenging time-continuous setting, and interpreted as motor task where 1 out of 10 fingers has to be bent upon seeing a corresponding visual stimulus (see Fig. 3a). In the context of our moving equilibrium hypothesis, we postulate that during the learning phase, but not the testing phase, an auditory signal identifying the correct finger sets the target spindle lengths of the 10 finger flexors, $\boldsymbol{u}_{\boldsymbol{o}}^*(t)$, i.e. a desired contraction for the correct finger in response to the visual input $\boldsymbol{r}_{\text{in}}(t)$, and a desired relaxation for the 9 incorrect fingers.

We train a hierarchical three-layer network on images of handwritten digits (MNIST, LeCun, 1998), with image presentation times between 0.5τ ($=5$ ms, with τ the membrane time constant) and 20τ ($=200$ ms). Fig. 4a-c1 depict the most challenging scenario with the shortest presentation time. Synaptic plasticity is continuously active, despite the network never reaching a steady state (Fig. 4b1). Due to the lookahead firing rates in NLA, the mismatch errors $\bar{e}_i(t)$ propagate without lag throughout the network, as explained above. As a consequence, our mismatch errors are equal to the errors obtained from classical error backpropagation applied at each time step to the purely forward network, without error-correcting the voltage, and instead, at each layer l considering only the forward voltage updates $\boldsymbol{u}_l = \boldsymbol{W}_l\,\rho(\boldsymbol{u}_{l-1})$, see Fig. 4b2. After learning, the network learned to implement the mapping $\boldsymbol{u}_{\boldsymbol{o}} = \boldsymbol{F}_W(\bar{\boldsymbol{r}}_{\text{in}}) \approx \boldsymbol{u}_{\boldsymbol{o}}^*$ with a performance comparable to classical error-backpropagation, despite the short presentation time of only 5 ms (Fig. 4c). The limiting presentation time of 5 ms is due to the low-pass filtering of the input rates $\boldsymbol{r}_{\text{in}}$ entering in $\boldsymbol{F}_W(\bar{\boldsymbol{r}}_{\text{in}})$, which we choose to be 10 ms. We also chose the membrane time constant of the other network neurons to be 10 ms, but these could have been much longer, without violating the instantaneous voltage propagation (as long as the look-ahead rates again compensate these longer time constants).

The instantaneous voltage propagation reduces an essential constraint of previous models of bio-plausible error backpropagation (e.g., Scellier & Bengio (2017), Whittington & Bogacz (2017), and Sacramento *et al.* (2018), with reviews Richards *et al.* (2019), Whittington & Bogacz (2019), and Lillicrap, Santoro, *et al.* (2020)): without lookahead firing rates, networks need much longer to correctly propagate errors across layers, with each layer roughly adding another membrane time constant of 10 ms, and thus cannot cope with realistic input presentation times. In fact, in networks without lookahead output, learning is only successful if plasticity is switched off while the network dynamics is not stationary (Fig. 4c2). As a comparison, neuronal response latency in the primary visual cortex (V1) of rats to flashing stimuli are in the order of 50 ms if the cortex is in a synchronized state, shortens to roughly 40 ms if in a desynchronized state (Wang *et al.,* 2014), and potentially shortens further if the area is preactivated through expectations (Blom *et al.,* 2020).

Implementation in cortical microcircuits

So far, we did not specify how errors $\boldsymbol{e}$ appearing in the differential equation for the voltage (Eq. 7a) are transmitted across the network in a biologically plausible manner. Building on Sacramento *et al.,* 2018, we propose a cortical microcircuit to enable this error transport, with all neuron dynamics evolving according to the NLA principle. Although the idea applies for arbitrarily connected networks, we use the simpler case of functionally feedforward networks to illustrate the flow of information in these microcircuits (Fig. 5a).

For such an architecture, pyramidal neurons in area l (that is a 'layer' of the feedforward network) are accompanied by a pool of interneurons in the same layer (area). The dendrites of the interneurons integrate in time (with time constant τ) lateral input from pyramidal neurons of the same layer ($\boldsymbol{r}_l$) through plastic weights $\boldsymbol{W}_l^{\text{IP}}$. Additionally, interneurons receive 'top-down nudging' from pyramidal neurons in the next layer through randomly initialized and fixed backprojecting synapses $\boldsymbol{B}_l^{\text{IP}}$ targeting the somatic region, and interneuron nudging strength β^{I}. The notion of 'top-down' originates from the functionally feed-forward architecture leading from sensory to 'higher cortical areas'. In the context of motor control, the highest 'area' is last stage controlling the muscle lengths, being at the same time the first stage for the proprioceptive input (Fig. 3a).

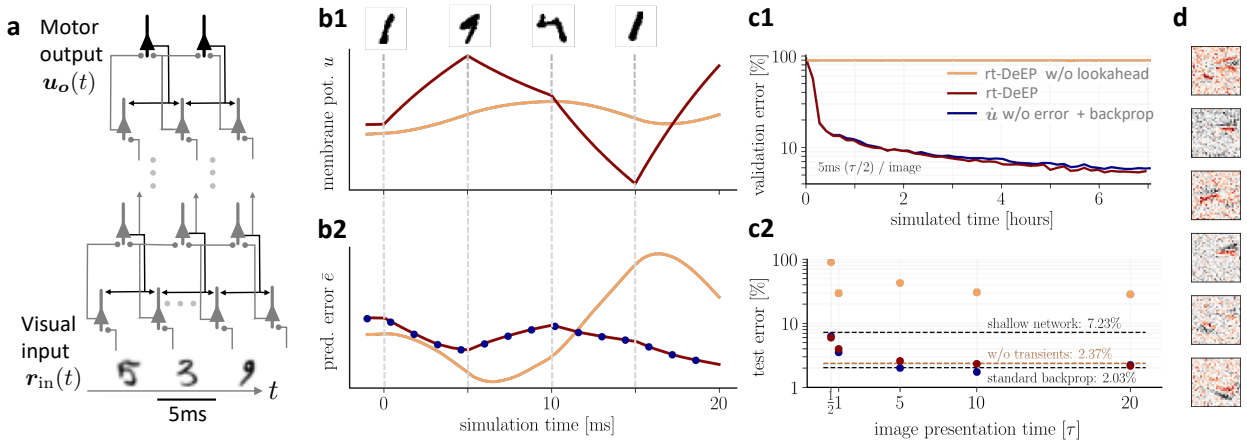

Figure 4: **On-the-fly learning of finger responses to visual input with real-time Dendritic Error Propagation (rt-DeEP). (a)** Functionally feedforward network with handwritten digits as visual input ($\boldsymbol{r}_{\text{in}}^{(2)}(t)$ in Fig. 3a, here from the MNIST data set, 5 ms presentation time per image), backprojections enabling credit assignment, and activity of the 10 output neurons interpreted as commands for the 10 fingers (forward architecture: $784 \times 500 \times 10$ neurons). **(b)** Example voltage trace (b1) and local error (b2) of a hidden neuron in NLA (red) compared to an equivalent network without lookahead rates (orange). Note that neither network achieves a steady state due to the extremely short input presentation times. Errors calculated via exact backpropagation, i.e. by using the error backpropagation algorithm on a purely feedforward NLA network at every simulation time step (with output errors scaled by β), shown for comparison (blue). **(c)** Comparison of network models during and after learning. Color scheme as in (b). **(c1)** The test error under NLA evolves during learning on par with classical error backpropagation. In contrast, networks without lookahead rates are incapable of learning such rapidly changing stimuli. **(c2)** With increasing presentation time, the performance under NLA further improves, while networks without lookahead rates stagnate at high error rates. This is caused by transient, but long-lasting misrepresentation of errors following stimulus switches: when plasticity is turned off during transients and is only active in the steady state, comparably good performance can be achieved (dashed orange). **(d)** Receptive fields of 6 hidden-layer neurons after training, demonstrating that even for very brief image presentation times (5ms), the combined neuronal and synaptic dynamics are capable of learning useful feature extractors such as edge filters.

According to the biophysics of the interneuron, the somatic membrane potential becomes a convex combination of the two types of afferent input (Urbanczik & Senn, 2014),

$$\boldsymbol{u}_l^{\text{I}} = (1-\beta^{\text{I}})\boldsymbol{W}_l^{\text{IP}}\bar{\boldsymbol{r}}_l + \beta^{\text{I}}\boldsymbol{B}_l^{\text{IP}}\boldsymbol{u}_{l+1} \; . \tag{9}$$

In the biological implementation, the feedback input is mediated by the low-pass filtered firing rates $\bar{\boldsymbol{r}}_{l+1} = \rho(\boldsymbol{u}_{l+1})$, not by $\boldsymbol{u}_{l+1}$ as expressed in the above equation. Yet, we argue that for a threshold-linear ρ the 'top-down nudging' by the rate $\bar{\boldsymbol{r}}_{l+1}$ is effectively reduced to a nudging by the voltage $\boldsymbol{u}_{l+1}$. This is because errors are only backpropagated when the slope of the transfer function is positive, $\boldsymbol{r}'_{l+1} > 0$, and hence when the upper-layer voltage is in the linear regime. For more general transfer functions, we argue that short-term synaptic depression may invert the low-pass filtered presynaptic rate back to the presynaptic membrane potential, $\bar{\boldsymbol{r}}_{l+1} \to \boldsymbol{u}_{l+1}$, provided that the recovery time constant τ matches the membrane time constant (see end of the section and Methods).

Apical dendrites of pyramidal neurons in each layer receive top-down input from the pyramidal population in the upper layer through synaptic weights $\boldsymbol{B}_l$. These top-down weights could be learned to predict the lower-layer activity (Rao & Ballard, 1999) or to become the transposed of the forward weight matrix (Max *et al.,* 2022), but for simplicity we randomly initialized them and keep them fixed (Lillicrap, Santoro, *et al.,* 2020). Beside the top-down projections the apical dendrites also receive lateral input via an interneuron population in the same layer through synaptic weights $-\boldsymbol{W}_l^{\text{PI}}$ that are plastic and will be learned to obtain suitable dendritic errors. The '-' sign is suggestive for these interneurons to subtract away the top-down input entering through $\boldsymbol{B}_l$ (while the weights can still be positive or negative). Assuming again a conversion of rates to voltages, also for the inhibitory neurons that may operate in a linear regime, the overall apical voltage becomes

$$\bar{\boldsymbol{e}}_l^A = \boldsymbol{B}_l\boldsymbol{u}_{l+1} - \boldsymbol{W}_l^{\text{PI}}\boldsymbol{u}_l^{\text{I}} \; . \tag{10}$$

What cannot be explained away from the top-down input $\boldsymbol{B}_l\boldsymbol{u}_{l+1}$ by the lateral feedback, $-\boldsymbol{W}_l^{\text{PI}}\boldsymbol{u}_l^{\text{I}}$, remains as dendritic prediction error $\bar{\boldsymbol{e}}_l^A$ in the apical tree (Fig. 5a). If the top-down and lateral feedback weights are learned as outlined next, these apical prediction errors take the role of the backpropagated errors in the classical backprop algorithm.

To adjust the interneuron circuit in each layer ('area'), the synaptic strengths from pyramidal-to-interneurons, $\boldsymbol{W}_l^{\mathrm{IP}}$, are learned to minimize the interneuron mismatch energy, $E_l^{\mathrm{IP}} = \frac{1}{2}\|\boldsymbol{u}_l^{\mathrm{I}} - \boldsymbol{W}_l^{\mathrm{IP}}\bar{\boldsymbol{r}}_l\|^2$. The interneurons, while being driven by the lateral inputs $\boldsymbol{W}_l^{\mathrm{IP}}\bar{\boldsymbol{r}}_l$, learn to reproduce the upper-layer activity that also nudges the interneuron voltage. Learning is accomplished if the upper-layer activity, in the absence of an additional error on the upper layer, is fully reproduced in the interneurons by the lateral input.

Once the interneurons learned to represent the 'error-free' upper-layer activity, they can be used to explain away the top-down activities that also project to the apical trees. The synaptic strengths from the inter-to-pyramidal neurons, $\boldsymbol{W}_l^{\mathrm{PI}}$, are learned to minimize the apical mismatch energy, $E_l^{\mathrm{PI}} = \frac{1}{2}\|\bar{\boldsymbol{e}}_l^A\|^2 = \frac{1}{2}\|\boldsymbol{B}_l\boldsymbol{u}_{l+1} - \boldsymbol{W}_l^{\mathrm{PI}}\boldsymbol{u}_l^{\mathrm{I}}\|^2$. While in the absence of an upper-layer error, the top-down activity $\boldsymbol{B}_l\boldsymbol{u}_{l+1}$ can be fully cancelled by the interneuron activity $\boldsymbol{W}_l^{\mathrm{PI}}\boldsymbol{u}_l^{\mathrm{I}}$, a neuron-specific error will remain in the apical dendrites of the lower-level pyramidal neurons if there was an error endowed in the upper-layer neurons. Gradient descent learning on these two energies results in the learning rules for the P-to-I and I-to-P synapses,

$$\dot{\boldsymbol{W}}_l^{\mathrm{IP}} = \eta^{\mathrm{IP}}(\boldsymbol{u}_l^{\mathrm{I}} - \boldsymbol{W}_l^{\mathrm{IP}}\bar{\boldsymbol{r}}_l)\,\bar{\boldsymbol{r}}_l^{\mathrm{T}} \quad \text{and} \quad \dot{\boldsymbol{W}}_l^{\mathrm{PI}} = \eta^{\mathrm{PI}}(\boldsymbol{B}_l\boldsymbol{u}_{l+1} - \boldsymbol{W}_l^{\mathrm{PI}}\boldsymbol{u}_l^{\mathrm{I}})\,\boldsymbol{u}_l^{\mathrm{I}^{\mathrm{T}}}\,. \tag{11}$$

The following theorem on dendritic error learning tells that the plasticity in the lateral feedback loop leads to an appropriate error representation in the apical dendrites of pyramidal neurons.

Theorem 2 (real-time Dendritic Error Learning, rt-DeEL) *Consider a cortical microcircuit composed of pyramidal and interneurons, as illustrated in Fig. 5a (with dimensionality constraints specified in Methods, and fixed or dynamic weights $\boldsymbol{W}_l^{IP}$ to the interneurons). Then the inter-to-pyramidal synapses evolve at each layer l ('cortical area') such that the lateral feedback circuit aligns with the top-down feedback circuit,*

$$\boldsymbol{W}_l^{PI}\boldsymbol{W}_l^{IP} = \boldsymbol{B}_l\boldsymbol{W}_{l+1}\,. \tag{12}$$

In the presence of an output nudging, the apical voltages of the layer-l pyramidal neurons (Eq. 10) then represent the 'B-backpropagated' error $\bar{\boldsymbol{e}}_l^A = \boldsymbol{B}_l\,\bar{\boldsymbol{e}}_{l+1}$. When modulated by the postsynaptic rate derivative, $\bar{\boldsymbol{r}}_l' = \boldsymbol{\rho}'(\bar{\boldsymbol{u}}_l)$, this apical error ensures the correct representation of errors $\bar{\boldsymbol{e}}_l$ for the real-time dendritic error propagation (rt-DeEP, Theorem 1),

$$\bar{\boldsymbol{e}}_l = \boldsymbol{u}_l - \boldsymbol{W}_l\bar{\boldsymbol{r}}_{l-1} = \bar{\boldsymbol{r}}_l'\cdot\bar{\boldsymbol{e}}_l^A = \bar{\boldsymbol{r}}_l'\cdot\boldsymbol{B}_l\,\bar{\boldsymbol{e}}_{l+1}\,. \tag{13}$$

Simultaneously learning apical errors and basal signals. Microcircuits following these neuronal and synaptic dynamics are able to learn the classification of hand-written digits from the MNIST dataset while learning the apical signal representation (Fig. 5b1-2). In this case, feedforward weights $\boldsymbol{W}_l$ and lateral weights $\boldsymbol{W}_l^{\mathrm{PI}}$ and $\boldsymbol{W}_l^{\mathrm{IP}}$ are all adapted simultaneously. Including the $\dot{\boldsymbol{W}}_l^{\mathrm{IP}}$-plasticity (by turning on the interneuron nudging from the upper layer, $\beta^{\mathrm{I}} > 0$ in Eq. 9), greatly speeds up the learning.

With and without $\dot{\boldsymbol{W}}_l^{\mathrm{IP}}$-plasticity, the lateral feedback via interneurons (with effective weight $\boldsymbol{W}_l^{\mathrm{IP}}\boldsymbol{W}_l^{\mathrm{PI}}$) learns to align with the forward-backward feedback via upper layer pyramidal neurons (with effective weight $\boldsymbol{B}_l\boldsymbol{W}_{l+1}$, Fig. 5b3). The microcircuit extracts the gradient-based errors (Eq. 13), while the forward weights use these errors to reduce these errors to first minimize the neuron-specific mismatch errors, and eventually the output cost.

Since the apical voltage $\bar{\boldsymbol{e}}_l^A$ appears as a postsynaptic factor in the plasticity rule for the interneurons ($\dot{\boldsymbol{W}}_l^{\mathrm{PI}}$), this I-to-P plasticity can be interpreted as Hebbian plasticity of inhbitory neurons, consistent with earlier suggestions (Vogels *et al.,* 2012; Bannon *et al.,* 2020). The plasticity $\dot{\boldsymbol{W}}_l^{\mathrm{IP}}$ of the P-to-I synapses, in the same way as the plasticity for the forward synapses $\dot{\boldsymbol{W}}_l$, can be interpreted as learning from the dendritic prediction of somatic activity (Urbanczik & Senn, 2014).

Crucially, choosing a large enough interneuron population, the simultaneous learning of the lateral microcircuit and the forward network can be accomplished without fine-tuning of parameters. As an instance in case, all weights shared the same learning rate. Such stability bolsters the biophysical plausibility of our NLA framework and improves over the previous, more heuristic approach (Sacramento *et al.,* 2018; Mesnard *et al.,* 2019). The stability may be related to the nested gradient descent learning according to which somatic and apical mismatch errors in pyramidal neurons, and somatic mismatch errors in inhibitory neurons are minimized.

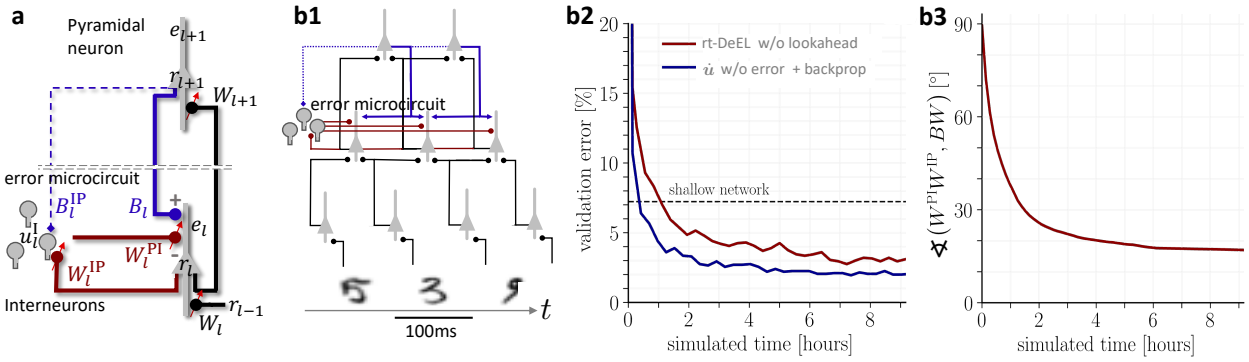

Figure 5: **Hierarchical plastic microcircuits implement real-time Dendritic Error Learning (rt-DeEL). (a)** Microcircuit with 'top-down' input (originating from peripheral motor activity, blue line) that is explained away by the lateral input via interneurons (dark red), with the remaining activity representing the error $\bar{e}_l$. Plastic connections are denoted with a small red arrow and nudging with a dashed line. **(b1)** Simulated network with 784-300-10 pyramidal-neurons and a population of 40 interneurons in the hidden layer used for the MNIST learning task where the handwritten digits have to be associated to the 10 fingers. **(b2)** Test errors for rt-DeEL with joint tabula rasa learning of the forward and lateral weights of the microcircuit. A similar performance is reached as with classical error backpropagation. For comparability, we also show the performance of a shallow network (dashed line). **(b3)** Angle derived from the Frobenius norm between the lateral pathway $\boldsymbol{W}_l^{\text{IP}}\boldsymbol{W}_l^{\text{PI}}$ and the feedback pathway $\boldsymbol{B}_l\boldsymbol{W}_{l+1}$. During training, both pathways align to allow correct credit assignment throughout the network. Indices are dropped in the axis label for readability.

Finally, since errors are defined at the level of membrane voltages (Eq. 11), synapses need a mechanism by which they can recover the presynaptic voltage errors from their afferent firing rates. While for threshold-linear transfer functions this is not too involved (Methods), more general neuronal nonlinearities must be matched by corresponding synaptic nonlinearities. Pfister *et al.* (2010) have previously illustrated how spiking neurons can leverage short-term synaptic depression to estimate the membrane potential of their presynaptic partners. Here, we assume a similar mechanism in the context of our rate-based neurons. The monotonically increasing neuronal activation function, $\bar{\boldsymbol{r}}_{l+1} = \rho(\boldsymbol{u}_{l+1})$, can be approximately compensated by a vesicle release probability that monotonically decreases with the low-pass filtered presynaptic rate $\bar{\boldsymbol{r}}_{l+1}$ (see SI and Fig. 6 therein). If properly matched, this leads to a linear relationship between the presynaptic membrane potential $\boldsymbol{u}_{l+1}$ and the postsynaptic voltage contribution.

Discussion

We introduced a least-action principle for neuronal networks from which we derived the membrane potential dynamics of the involved neurons, together with gradient descent plasticity of the synapses in the network. The central notion of the theory is the somato-dendritic mismatch error in each individual neuron. Given the various findings on the dendritic integration of top-down signals (Larkum, 2013; Takahashi *et al.,* 2020), we suggest that the error is formed in the apical dendrite of pyramidal neurons. Active dendritic and somatic processes compensate for delays caused by the dendritic error propagation and the somatic integration. This leads to the prospective firing of pyramidal neurons that are jointly driven by the forward input on the basal dendrites and the error on the apical dendrite. We derived a local plasticity rule for synapses on the basal tree that, by extracting the apical feedback error, reduces the prospective somato-dendritic mismatch error in an individual neuron. This plasticity is shown to also globally reduce the instantaneous cost at output neurons (defined by deviations from target voltages) at any moment in time while stimuli and targets may continuously change. Motor control offers an intuitive context where self-correcting prospective networks may enter. We put forward the moving equilibrium hypothesis, according to which sensory input, motor commands and muscle spindle feedback are in a recurrent equilibrium at any moment during the movement.

We further showed how a local microcircuit can serve to calculate a neuron-specific error in the apical tree of each individual pyramidal neuron. The apical tree of pyramidal neurons receives feedback from higher-level neurons in the network, while local inhibitory neurons try on the fly to 'explain away' the top-down expectations. What cannot be explained away remains as apical prediction error. This prediction error eventually induces error-correcting plasticity at the sensory-driven input on the basal tree. To find the correct error representation in the apical tree, plasticity of the inter-to-pyramidal neurons seeks to homeostatically drive the apical de- or hyperpolarized voltage back to rest.

While the network synapses targetting the basal tree are performing gradient descent on the global cost, the microcircuit synapses involved in the lateral feedback are gradient descent on local error functions, both at any moment in time.

Our work builds on three general lines of research on formalizing the learning capabilities of cortical networks. The first line refers to the use of an energy function to jointly infer the neuronal dynamics and synaptic plasticity, originally formulated for discrete-time networks (Hopfield, 1982; Ackley *et al.,* 1985), and recently extended to continuous-time networks (Scellier & Bengio, 2017). The second line refers to understanding error-backpropagation in the brain (Rumelhart *et al.,* 1986; Xie & Seung, 2003; Whittington & Bogacz, 2017; Whittington & Bogacz, 2019; Lillicrap, Santoro, *et al.,* 2020). The third line refers to the use of dendritic compartmentalization in various kinds of computation (Schiess *et al.,* 2016; Poirazi & Papoutsi, 2020), recently linked to deep learning (Guerguiev *et al.,* 2017; Sacramento *et al.,* 2018; Haider *et al.,* 2021).

With regard to energy functions, the NLA principle adds a variational approach to characterize continuous-time neuronal trajectories and plasticity. Variational approaches are studied in the context of optimal control theory where a cost integral is minimized across time, constrained to some network dynamics (Todorov & Jordan, 2002; Meulemans, Farinha, *et al.,* 2021). The NLA represents a unifying notion that allows to infer both, the network dynamics and its optimal control from a single Lagrangian. The error we derive represents prospective control variables that are applied to the voltages of each network neurons so that they push the output neurons towards their target trajectory. The full expression power of this control theoretic framework has yet to be proven when it is extended to genuine temporal processing that includes longer time constants, for instance inherent in a slow threshold adaptation (Bellec *et al.,* 2020). The NLA principle can also treat the case of strong feedback studied so far in relaxation networks only (Meulemans, Zucchet, *et al.,* 2022; Song *et al.,* 2022). Our rt-DeEP Theorem makes a statement for real-time gradient descent learning while the network is in a moving equilibrium, linking to motor learning in the presence of perturbing force fields (Herzfeld *et al.,* 2014) or perturbing visual inputs (Dimitriou, 2016).

With regard to error-backpropagation, the NLA principle relies on feedback alignment (Lillicrap, Cownden, *et al.,* 2016) to correctly interpret the apical errors. Other works have explored learning of feedback weights (Akrout *et al.,* 2019; Kunin *et al.,* 2020; Max *et al.,* 2022). Since these are complementary to the principles suggested here, a promising direction would be to explore how feedback weights can be learned in NLA microcircuits to improve credit assignment in deeper networks.

With regard to dendritic and neuronal processing, we emphasize the prospective nature of apical voltages and somatic firing (Fig. 2). The prospective coding of errors and signals overcomes the various delays inherent in our previous approach (Sacramento *et al.,* 2018; Mesnard *et al.,* 2019). Each neuron of the network instantaneously corrects its voltage so that the output neurons are pushed towards the target trajectories. Ongoing synaptic plasticity adjusts the synaptic strengths so that the errors in each neuron are reduced at any moment in time, without needing to wait for network relaxations (Scellier & Bengio, 2017; Song *et al.,* 2022). Yet, because in the present framework the input rates are still low-pass filtered, step inputs cannot be instantaneously propagated, only their low-pass filterings. Haider *et al.,* 2021 offers a framework that is also suited instantaneously process step inputs.

Motivated by the predictive power of the least-action principle in physics, we ask about experimental confirmation and predictions of the NLA in biology. Given its axiomatic approach, it appears astonishing to find various preliminary matches at the dendritic, somatic, interneuron, synaptic and even behavioural level. Some of these are: (1) the prospective coding of pyramidal neuron firing (Köndgen *et al.,* 2008); (2) the prospective processing of apical signals while propagating to the soma (Ulrich, 2002); (3) the basal synaptic plasticity on pyramidal neurons and synaptic plasticity on interneurons, driven by the postsynaptic activity that is 'unexplained' by the distal dendritic voltage (Urbanczik & Senn, 2014); (4) the Hebbian homeostatic plasticity of interneurons targeting the apical dendritic tree of pyramidal neurons (Bannon *et al.,* 2020); (5) the short-term synaptic depression at top-down synapses targetting inhibitory neurons and apical dendrites (akin to Abbott *et al.,* 1997, but with a faster recovery time constant) that invert the presynaptic activation function (see also Pfister *et al.,* 2010); (6) the modulation of the apical contribution to the somatic voltage by the slope of the somatic activation function (for instance by downregulating apical NMDA receptors with increasing rate of backpropagating action potentials, Theis *et al.,* 2018); and (7) the role of muscle spindles in the prospective encoding of motor errors during motor learning (Dimitriou, 2016; Papaioannou & Dimitriou, 2021). More theoretical and experimental work is required to explore these various links.

Overall, our approach introduces a method from theoretical physics to computational neuroscience and couples it with a normative perspective on the dynamical processing of neurons and synapses within global cortical networks and local microcircuits. Given its physical underpinnings, the approach may inspire the rebuilding of computational principles of cortical neurons and circuits in neuromorphic hardware (Bartolozzi *et al.,* 2022). A step in this direction, building on the instantaneous computational capabilities with slowly integrating neurons, has been done (Haider *et al.,* 2021) and a next challenge is to generalize the NLA principle to spiking neurons (Zenke & Ganguli, 2018; Göltz *et al.,* 2021) and longer temporal processing.

Acknowledgments

We would like to thank Federico Benitez, Jonathan Binas, Paul Haider, Kevin Max, Alexander Mathis, and Alexander Meulemans and Jean-Pascal Pfister for helpful discussions, Kaspar Schindler for providing the human intracortical EEG data, and the late Karlheinz Meier for his dedication and support throughout the early stages of the project. WS particularly thanks Nicolas Zucchet for inspiring mathematical discussions and hints to literature on time-varying optimal control. This work has received funding from the European Union 7th Framework Programme under grant agreement 604102 (HBP), the Horizon 2020 Framework Programme under grant agreements 720270, 785907 and 945539 (HBP) and the Swiss National Science Foundation (SNSF, Sinergia grant CRSII5180316; João Sacramento is supported by SNSF Ambizione grant PZ00P3_186027). We further express our particular gratitude towards the Manfred Stärk Foundation for their continued support. We acknowledge the use of Fenix Infrastructure resources, which are partially funded from the European Union's Horizon 2020 research and innovation programme through the ICEI project under the grant agreement No. 800858.

Methods

Euler-Lagrange equations as inverse low-pass filters

The theory is based on the lookahead of neuronal quantities. In general, the lookahead of a trajectory $x(t)$ is defined as $\mathscr{L}x = x + \tau\dot{x}$, with lookahead operator

$$\mathscr{L} := 1 + \tau\frac{\mathrm{d}}{\mathrm{d}t}\,. \tag{14}$$

The lookahead operator is the inverse of the low-pass filter operator denoted by a bar,

$$\bar{x}(t) := \frac{1}{\tau}\int_{-\infty}^{t} x(t')e^{-\frac{t-t'}{\tau}}\mathrm{d}t'\,. \tag{15}$$

This low-pass filtering can also be characterized by the differential equation $\tau\dot{\bar{x}}(t) = -\bar{x}(t) + x(t)$, see SI. Hence, applying the low-pass filtering to x and then the lookahead operator $\mathscr{L}$ to $\bar{x}(t)$, and using the Leibnitz rule for differentiating an integral, we calculate $\mathscr{L}\bar{x}(t) = x(t)$. In turn, applying first the lookahead, and then the low-pass filtering, also yields the original trace back, $\overline{\mathscr{L}x} = x$ (see SI).

We consider an arbitrary network architecture with network neurons that are recurrently connected and that receive external input through an overall weight matrix $\boldsymbol{W} = (\boldsymbol{W}_{\text{in}}, \boldsymbol{W}_{\text{net}})$, aggregated column-wise. The instantaneous presnyaptic firing rates are $\boldsymbol{r} = (\boldsymbol{r}_{\text{in}}, \boldsymbol{r}_{\text{net}})$, interpreted as a single column vector. A subset of network neurons are output neurons, $\mathcal{O} \subseteq \mathcal{N}$, for which target voltages $\boldsymbol{u}^*$ may be imposed. Rates and voltages may change in time t. Network neurons are assigned a voltage $\boldsymbol{u}$, generating the low-pass filtered rate $\bar{\boldsymbol{r}}_{\text{net}} = \rho(\boldsymbol{u})$, and a low-pass filtered error $\bar{\boldsymbol{e}} = \boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}}$. We further define output errors $\bar{e}_o^* = u_o^* - u_o$ for $o \in \mathcal{O}$, and $\bar{e}_i^* = 0$ for non-output neurons $i \in \mathcal{N} \setminus \mathcal{O}$. With this, the Lagrangian from Eq. 3 takes the form

$$L = \frac{1}{2}\|\bar{\boldsymbol{e}}\|^2 + \frac{\beta}{2}\|\bar{\boldsymbol{e}}^*\|^2\,. \tag{16}$$

We next use that $\boldsymbol{u} = \tilde{\boldsymbol{u}} - \tau\dot{\tilde{\boldsymbol{u}}}$, with the $\tilde{\cdot}$ operator defined in Eq. 4, to write out the Lagrangian L in the canonical coordinates $(\tilde{\boldsymbol{u}}, \dot{\tilde{\boldsymbol{u}}})$ as (see also Eq. 3)

$$L = \frac{1}{2}\sum_{i\in\mathcal{N}}\Big[\tilde{u}_i - \tau\dot{\tilde{u}}_i - \textstyle\sum_j W_{ij}\rho(\tilde{u}_j - \tau\dot{\tilde{u}}_j)\Big]^2 + \frac{\beta}{2}\sum_{o\in\mathcal{O}}\big[u_o^* - (\tilde{u}_o - \tau\dot{\tilde{u}}_o)\big]^2\,. \tag{17}$$

The neuronal dynamics is derived from requiring a stationary action (see Eq. 5), which is generally solved by the Euler-Lagrange equations $\frac{\partial L}{\partial \tilde{u}_i} - \frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{\tilde{u}}_i} = 0$ (see Eq. 6). Because $\tilde{u}$ only arises in L in the compound $\tilde{u} - \tau\dot{\tilde{u}}$, the derivative of L with respect to $\tilde{u}$ is identical to the derivative with respect to $\tau\dot{\tilde{u}}$,

$$\frac{\partial L}{\partial \dot{\tilde{u}}_i} = -\tau\frac{\partial L}{\partial \tilde{u}_i}\,. \tag{18}$$

Using the lookahead operator Eq. 14, the Euler-Lagrange equations can then be rewritten as

$$\frac{\partial L}{\partial \tilde{u}_i} + \tau\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \tilde{u}_i} = \mathscr{L}\frac{\partial L}{\partial \tilde{u}_i} = 0\,. \tag{19}$$

Since $L(\tilde{\boldsymbol{u}}, \dot{\tilde{\boldsymbol{u}}}) = L(\boldsymbol{u})$ and $\boldsymbol{u} = \tilde{\boldsymbol{u}} - \tau\dot{\tilde{\boldsymbol{u}}}$, the derivative of L with respect to $\tilde{\boldsymbol{u}}$ is the same as the derivative of L with respect to $\boldsymbol{u}$, $\frac{\partial L}{\partial \tilde{u}_i} = \frac{\partial L}{\partial u_i}$. Plugging this into Eq. 19, the Euler-Lagrange equations become a function of $\boldsymbol{u}$ and $\dot{\boldsymbol{u}}$,

$$\mathscr{L}\frac{\partial L}{\partial u_i} = 0\,. \tag{20}$$

The solution of this differential equation is $\frac{\partial L}{\partial u_i} = c_i\, e^{-\frac{t-t_0}{\tau}}$, and hence any trajectory $(\tilde{u}_i, \dot{\tilde{u}}_i)$ which satisfy the Euler-Lagrange equations will hence cause $\frac{\partial L}{\partial u_i}$ to converge to zero with a characteristic time scale of τ. Since we require that the initialisation is at $t_0 = -\infty$, we conclude that $\frac{\partial L}{\partial u_i} = 0$, as required in the rt-DeEP Theorem.

Deriving the network dynamics from the Euler-Lagrange equations

We now derive the equations of motion from the Euler-Lagrange equations. Noticing that $\boldsymbol{u}$ enters in $\bar{\boldsymbol{e}} = \boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}}$ twice, directly and through $\bar{\boldsymbol{r}}_{\text{net}} = \rho(\boldsymbol{u})$, and once in the output error $\bar{\boldsymbol{e}}^*$, we calculate from Eq. 16, using $\bar{\boldsymbol{r}}(\boldsymbol{u}) = (\bar{\boldsymbol{r}}_{\text{in}}, \rho(\boldsymbol{u}))$ and $\boldsymbol{W} = (\boldsymbol{W}_{\text{in}}, \boldsymbol{W}_{\text{net}})$,

$$\frac{\partial L}{\partial \boldsymbol{u}} = \bar{\boldsymbol{e}} - \bar{\boldsymbol{\epsilon}} - \beta\bar{\boldsymbol{e}}^*\,, \quad \text{with} \quad \bar{\boldsymbol{\epsilon}} = \bar{\boldsymbol{r}}'_{\text{net}} \cdot \boldsymbol{W}_{\text{net}}^{\mathrm{T}} \bar{\boldsymbol{e}}\,. \tag{21}$$

Next, we apply the lookahead operator to this expression, as required by the Euler-Lagrange equations Eq. 19. In general $\mathscr{L}\bar{\boldsymbol{x}} = \bar{\boldsymbol{x}} + \tau\dot{\bar{\boldsymbol{x}}} = x$, and we set for $\bar{\boldsymbol{x}}$ the expression on the right-hand side of Eq. 21, $\bar{\boldsymbol{x}} = \bar{\boldsymbol{e}} - \bar{\boldsymbol{\epsilon}} - \beta\bar{\boldsymbol{e}}^*$, which at the same time is $\bar{\boldsymbol{x}} = \frac{\partial L}{\partial \boldsymbol{u}}$. Hence, the Euler-Lagrange equations in the form of Eq. 20, $\mathscr{L}\bar{x} = 0$, translate into

$$\mathscr{L}\frac{\partial L}{\partial \boldsymbol{u}} = 0 \iff \boldsymbol{e} - \boldsymbol{\epsilon} - \beta\boldsymbol{e}^* = 0 \iff \tau\dot{\boldsymbol{u}} = -\boldsymbol{u} + \boldsymbol{W}\boldsymbol{r} + \boldsymbol{e}\,. \tag{22}$$

To move from the middle to the last equality we replaced $\boldsymbol{e}$ by $\boldsymbol{e} = \mathscr{L}\bar{\boldsymbol{e}} = \boldsymbol{u} + \tau\dot{\boldsymbol{u}} - \boldsymbol{W}\boldsymbol{r}$. In the last equality we interpret $\boldsymbol{e}$ as the sum of the two errors, $\boldsymbol{e} = \boldsymbol{\epsilon} + \beta\boldsymbol{e}^*$, again using the middle equality. This proves Eq. 7.

Notice that the differential equation $\tau\dot{\boldsymbol{u}} = ...$ in Eq. 22 represents an implicit ordinary differential equation as on the right-hand side not only $\boldsymbol{u}$, but also $\dot{\boldsymbol{u}}$ appears (in $\boldsymbol{r}$ and $\boldsymbol{e}$). The uniqueness of the solution $\boldsymbol{u}(t)$ for a given initial condition is only guaranteed if it can be converted into an explicit ordinary differential equation (see Sect. C).

In taking the temporal derivative we assumed small learning rates such that terms including $\dot{W}_{ij}$ can be neglected. The derived dynamics for the membrane potential of a neuron u_i in Eq. 22 show the usual leaky behavior of biological neurons. However, both presynaptic rates $\bar{r}_i$ and prediction errors $\bar{e}_i$ enter the equation of motion with lookaheads, i.e., they are advanced ($r_i = \bar{r}_i + \tau\dot{\bar{r}}_i$ and $e_i = \bar{e}_i + \tau\dot{\bar{e}}_i$), cancelling the low-pass filtering. Since $\dot{\bar{r}}_i = \rho'(u_i)\,\dot{u}_i$, the rate and error, r_i and e_i, can also be seen as nonlinear extrapolations from the voltage and its derivative into the future.

The instantaneous transmission of information throughout the network at the level of the voltages can now be seen by low-pass filtering Eq. 22 with initialization far back in the past,

$$\boldsymbol{u} = \overline{\boldsymbol{u} + \tau\dot{\boldsymbol{u}}} = \overline{\boldsymbol{W}\boldsymbol{r} + \boldsymbol{e}} = \boldsymbol{W}\bar{\boldsymbol{r}}(\boldsymbol{u}) + \bar{\boldsymbol{e}}\,, \tag{23}$$

with $\bar{\boldsymbol{r}}(\boldsymbol{u}) = (\bar{\boldsymbol{r}}_{\text{in}}, \rho(\boldsymbol{u}))$ interpreted as column vector, and $\bar{\boldsymbol{e}} = \bar{\boldsymbol{r}}'_{\text{net}} \cdot \boldsymbol{W}_{\text{net}}^{\mathrm{T}} \bar{\boldsymbol{e}} + \beta\bar{\boldsymbol{e}}^*$. Hence, solving the voltage dynamics for $\boldsymbol{u}$ (Eq. 7a), with apical voltage $\boldsymbol{e} = \bar{\boldsymbol{e}} + \tau\dot{\bar{\boldsymbol{e}}}$ derived from Eq. 7b, yields the somatic voltage $\boldsymbol{u}$ satisfying the self-consistency equation (Eq. 23) at any time. In other words, $\boldsymbol{u}$ and $\bar{\boldsymbol{e}}$ 'propagate instantaneously'.

Deriving the error backpropagation formula

For clarity, we derive the error backpropagation algorithm for layered networks here. These can be seen as a special case of a general network with membrane potentials $\boldsymbol{u}$ and all-to-all weight matrix $\boldsymbol{W}$ (as introduced in appendix H), where the membrane potentials decompose into layerwise membrane potential vectors $\boldsymbol{u}_l$ and the weight matrix into according block diagonal matrices $\boldsymbol{W}_l$ (with $\boldsymbol{W}_l$ being the weights that project into layer l).

Assuming a network with N layers, by low-pass filtering the equations of motion we get

$$\boldsymbol{u}_l = \boldsymbol{W}_l\bar{\boldsymbol{r}}_{l-1} + \bar{\boldsymbol{e}}_l\,, \tag{24}$$

$\forall l \in [1, N]$, with the output error $\bar{\boldsymbol{e}}_N = \beta\bar{\boldsymbol{e}}^* = \beta\,(\boldsymbol{u}_N^* - \boldsymbol{u}_N)$. The error $\boldsymbol{e} = \boldsymbol{\epsilon} + \beta\boldsymbol{e}^*$ we obtain from the general dynamics with $\bar{\boldsymbol{\epsilon}} = \bar{\boldsymbol{r}}'_{\text{net}} \cdot \boldsymbol{W}_{\text{net}}^{\mathrm{T}} \bar{\boldsymbol{e}}$, see Eq. 21 and Eq. 22, translates to an iterative formula for the error at the current layer l given the error at the downstream layer $l+1$, inherited from the drive $\bar{\boldsymbol{r}}_l = \rho(\boldsymbol{u}_l)$ of that downstream layer via $\boldsymbol{W}_{l+1}$,

$$\bar{\boldsymbol{e}}_l = \bar{\boldsymbol{r}}'_l \cdot \boldsymbol{W}_{l+1}^{\mathrm{T}} \bar{\boldsymbol{e}}_{l+1} \quad \text{for} \quad l < N\,. \tag{25}$$

and $\bar{\boldsymbol{e}}_N = \beta\,\bar{\boldsymbol{e}}^*$ for the output layer. The learning rule that reduces $\bar{\boldsymbol{e}}_l$ by gradient descent is proportional to this error and the presynaptic rate, as stated by Theorem 1, is

$$\dot{\boldsymbol{W}}_l \propto \left(\boldsymbol{u}_l - \boldsymbol{W}_l\,\bar{\boldsymbol{r}}_{l-1}\right)\bar{\boldsymbol{r}}_{l-1}^{\mathrm{T}} = \bar{\boldsymbol{e}}_l\,\bar{\boldsymbol{r}}_{l-1}^{\mathrm{T}}\,, \tag{26}$$

for $l = 1...N$. Eq. 25 and Eq. 26 together take the form of the error backpropagation algorithm, where an output error is iteratively propagated through the network and used to adjust the weights in order to reduce the output cost C. From this, it is easy to see that without output nudging (i.e., $\beta = 0$), the output error vanishes and consequently all other prediction errors vanish as well, $\bar{\boldsymbol{e}}_l = \boldsymbol{u}_l - \boldsymbol{W}_l\bar{\boldsymbol{r}}_l = 0$ for all $l \leq N$. This also means that in the absence of nudging, no weight updates are performed by the plasticity rule.

The learning rule for arbitrary connectivities is obtained in the same way by dropping the layer-wise notation. In this case, low-pass filtering the equations of motion yields $\boldsymbol{u} = \boldsymbol{W}\bar{\boldsymbol{r}} + \bar{\boldsymbol{e}}$, as calculated in Eq. 23, and the low-pass filtered error $\bar{\boldsymbol{e}} = \bar{\boldsymbol{\epsilon}} + \beta\bar{\boldsymbol{e}}^* = \bar{\boldsymbol{r}}'_{\text{net}}\cdot\boldsymbol{W}_{\text{net}}^{\mathrm{T}}\bar{\boldsymbol{e}} + \beta\bar{\boldsymbol{e}}^*$, as inferred from Eqs 21 and 22. Hence, the plasticity rule in general reads

$$\dot{\boldsymbol{W}} \propto \left(\boldsymbol{u} - \boldsymbol{W}\,\bar{\boldsymbol{r}}\right)\bar{\boldsymbol{r}}^{\mathrm{T}} = \bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^{\mathrm{T}}\,, \quad \text{with } \bar{\boldsymbol{e}} = \bar{\boldsymbol{r}}'_{\text{net}}\cdot\boldsymbol{W}_{\text{net}}^{\mathrm{T}}\bar{\boldsymbol{e}} + \beta\bar{\boldsymbol{e}}^*\,. \tag{27}$$

Proving Theorem 1 (rt-DeEP)

The implicit assumption in Theorem 1 is that $\dot{\boldsymbol{u}}$ exists in the distributional sense for $t > -\infty$, which is the case for delta-functions in $\boldsymbol{r}_{\text{in}}$ and step-functions in $\boldsymbol{u}^*$. Both parts (i) and (ii) of the Theorem are based on the requirement of stationary action $\delta A = 0$, and hence on $\boldsymbol{u}$ satisfying the Euler-Lagrange equations in the form of Eq. 22, $\mathscr{L}\frac{\partial L}{\partial \boldsymbol{u}} = 0$. From the solution $\frac{\partial L}{\partial u_i} = c\,e^{-\frac{t-t_0}{\tau}}$ we conclude that for initialization at $t_0 = -\infty$ we have $\frac{\partial L}{\partial \boldsymbol{u}} = 0$ for all t. It is the latter stronger condition that we require in the proof. With this, the main ingredient of the proof follows is the mathematical argument of Scellier & Bengio, 2017, according to which the total and partial derivative of L with respect to $\boldsymbol{W}$ are identical, and this in our case is true for any time t,

$$\frac{\mathrm{d}L}{\mathrm{d}\boldsymbol{W}} = \frac{\partial L}{\partial \boldsymbol{u}}^{\mathrm{T}}\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}\boldsymbol{W}} + \frac{\partial L}{\partial \boldsymbol{W}} = \frac{\partial L}{\partial \boldsymbol{W}}\,, \tag{28}$$

For convenience we considered $\frac{\partial L}{\partial \boldsymbol{u}}$ to be a column vector, deviating from the standard notations (see tutorial end of SI). Analogously to Eq. 28, we infer $\frac{\mathrm{d}L}{\mathrm{d}\beta} = \frac{\partial L}{\partial \beta}$. Reading Eq. 28 from the right to the left, we conclude that the learning rule $\dot{\boldsymbol{W}} \propto -\frac{\partial L}{\partial \boldsymbol{W}} = \bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^{\mathrm{T}}$ for all $\beta > 0$ is gradient descent on L, i.e. $\dot{\boldsymbol{W}} \propto -\frac{\mathrm{d}L}{\mathrm{d}\boldsymbol{W}}$. This total derivative of L can be analyzed for large and small β.

(i) We show that in the limit of large β, $\dot{\boldsymbol{W}}$ becomes gradient descent on the mismatch energy $E^{\mathrm{M}} = \frac{1}{2}\|\bar{\boldsymbol{e}}\|^2$. For this we first show that there is a solution of the self-consistency equation $\boldsymbol{u} = \boldsymbol{F}(\boldsymbol{u}) = \boldsymbol{W}\bar{\boldsymbol{r}} + \bar{\boldsymbol{r}}'_{\text{net}}\cdot\boldsymbol{W}_{\text{net}}^{\mathrm{T}}\bar{\boldsymbol{e}} + \beta\,\bar{\boldsymbol{e}}^*$ that is uniformly bounded for all t and β. For this we assume that the transfer function $\rho(u)$ is non-negative, monotonically increasing and bounded, that its derivative $\rho'(u)$ is bounded too, and that the input rates $\boldsymbol{r}_{\text{in}}$ and the target potentials $\boldsymbol{u}^*_{\boldsymbol{o}}$ are also uniformly bounded. To show that under these conditions we always find an uniformly bounded solution $\boldsymbol{u}(t)$, we first consider the case where the output voltages are clamped to the target, $\boldsymbol{u}_{\boldsymbol{o}} = \boldsymbol{u}^*_{\boldsymbol{o}}$ such that $\bar{\boldsymbol{e}}^* = 0$. For simplicity we assume that $\rho'(u) = 0$ for $|u| \geq c_0$. For voltages $\boldsymbol{u}$ with $\boldsymbol{u}_i \leq c_0$ the recurrent input current $\boldsymbol{W}\bar{\boldsymbol{r}}$ is bounded, say $|(\boldsymbol{W}\bar{\boldsymbol{r}})_j| \leq c_1$ for some $c_1 > c_0$. When including the error term $\bar{\boldsymbol{r}}'_{\text{net}}\cdot\boldsymbol{W}_{\text{net}}^{\mathrm{T}}\bar{\boldsymbol{e}}$, the total current still remains uniformly bounded, say $|\boldsymbol{F}(\boldsymbol{u})_j| \leq c_2$ for all $\boldsymbol{u}$ with $\boldsymbol{u}_i \leq c_0$. Because for larger voltages $\boldsymbol{u}_i > c_0$ the error term vanishes due to a vanishing derivative $\rho'(\boldsymbol{u}_i) = 0$, the mapping $\boldsymbol{F}(\boldsymbol{u})$ maps the c_2-box $\boldsymbol{u}$ (for which $|\boldsymbol{u}_i| \leq c_2$) onto itself. Brouwer's fixed point theorem then tells us that there is a fixed point $\boldsymbol{u} = \boldsymbol{F}(\boldsymbol{u})$ within the c_2-box. The theorem requires the continuity of $\boldsymbol{F}$, and this is assured if the neuronal transfer function $\bar{r} = \rho(u)$ is continuous.

We next relax the voltages of the output neurons from their clamped stage, $\boldsymbol{u}_{\boldsymbol{o}} = \boldsymbol{u}^*_{\boldsymbol{o}}$. Remember that these voltages satisfy $\boldsymbol{u}_{\boldsymbol{o}} = (\boldsymbol{W}\bar{\boldsymbol{r}} + \bar{\boldsymbol{r}}'_{\text{net}}\cdot\boldsymbol{W}_{\text{net}}^{\mathrm{T}}\bar{\boldsymbol{e}} + \beta\,\bar{\boldsymbol{e}}^*)_{\boldsymbol{o}} = \boldsymbol{F}(\boldsymbol{u})_{\boldsymbol{o}}$ at any time t. We determine the correction term $\beta\,\bar{\boldsymbol{e}}^*_{\boldsymbol{o}}$ such that in the limit $\beta \to \infty$ we get $\boldsymbol{u}_{\boldsymbol{o}} = \boldsymbol{F}(\boldsymbol{u})_{\boldsymbol{o}} = \boldsymbol{u}^*_{\boldsymbol{o}}$. The correction remains finite, and in the limit must be equal to $\lim_{\beta\to\infty}\beta\,\bar{\boldsymbol{e}}^*_{\boldsymbol{o}} = \boldsymbol{u}^*_{\boldsymbol{o}} - (\boldsymbol{W}\bar{\boldsymbol{r}} + \bar{\boldsymbol{r}}'_{\text{net}}\cdot\boldsymbol{W}_{\text{net}}^{\mathrm{T}}\bar{\boldsymbol{e}})_{\boldsymbol{o}}$. For arbitrary large nudging strength β, the output voltage $\boldsymbol{u}_{\boldsymbol{o}}$ deviates arbitrary little from the target voltage, $\boldsymbol{u}_{\boldsymbol{o}} = \boldsymbol{u}^*_{\boldsymbol{o}} + o(1/\beta)$, with target error $\bar{\boldsymbol{e}}^*_{\boldsymbol{o}} = \frac{1}{\beta}\left(\boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}} - \bar{\boldsymbol{r}}'_{\text{net}}\cdot\boldsymbol{W}_{\text{net}}^{\mathrm{T}}\bar{\boldsymbol{e}}\right)_{\boldsymbol{o}}$ shrinking like c_2/β. Likewise, also for non-output neurons i, the self-consistency solution $\boldsymbol{u}_i = \boldsymbol{F}(\boldsymbol{u})_i$ deviates arbitrarily little from the solution of the clamped state. To ensure the smooth drift of the fixed point while $1/\beta$ deviates from 0 we require that the Jacobian of $\boldsymbol{F}$ at the fixed point is invertible.

Because the output $\bar{\boldsymbol{e}}^*_{\boldsymbol{o}}$ shrinks with $1/\beta$, the cost shrinks quadratically with increasing nudging strength, $C = \frac{1}{2}\|\bar{\boldsymbol{e}}^*\|^2 = o\left(\frac{1}{\beta^2}\right)$, and hence the cost term $\frac{\beta}{2}\|\bar{\boldsymbol{e}}^*\|^2$ that enters in $L = E^{\mathrm{M}} + \frac{\beta}{2}\|\bar{\boldsymbol{e}}^*\|^2$ vanishes in the limit $\beta \to \infty$. In this large β limit, where $\bar{\boldsymbol{e}}^*_{\boldsymbol{o}} = 0$ and hence the outputs are clamped, $\boldsymbol{u}_{\boldsymbol{o}} = \boldsymbol{u}^*_{\boldsymbol{o}}$, the Lagrangian reduces to the mismatch energy, $L = E^{\mathrm{M}}$. Along the least-action trajectories we therefore get $\dot{\boldsymbol{W}} \propto -\frac{\partial L}{\partial \boldsymbol{W}} = -\frac{dL}{d\boldsymbol{W}} = -\frac{\mathrm{d}E^{\mathrm{M}}}{\mathrm{d}\boldsymbol{W}}$. The first equality uses

Eq. 28, and the second uses $L = E^{\mathrm{M}}$ just derived for $\beta = \infty$. This is statement (i) of Theorem 1. In the case of successful learning, $E^{\mathrm{M}} = 0$, we also conclude that the cost vanishes, $C = 0$. This is the case because $E^{\mathrm{M}} = 0$ implies $E^{\mathrm{M}}_o = 0$ for all output neurons o. Since $E^{\mathrm{M}}_o = \frac{1}{2}\bar{\boldsymbol{e}}^2_o = \frac{1}{2}(\bar{\boldsymbol{r}}'_{\mathrm{net}} \cdot \boldsymbol{W}^{\mathrm{T}}_{\mathrm{net}}\bar{\boldsymbol{e}} + \beta\,\bar{\boldsymbol{e}}^*)^2_o$, we conclude that $\bar{\boldsymbol{e}}_o = 0$, and if the output neurons do not feed back to the network (which we can assume without loss of generality), we conclude that $\bar{\boldsymbol{e}}^*_o = 0$.

(ii) To consider the case of small β, we use that the cost C can be expressed as $C = \frac{\partial L}{\partial \beta}$. This is a direct consequence of how C enters in $L = \frac{1}{2}\|\bar{\boldsymbol{e}}\|^2 + \frac{\beta}{2}C$, see Eq. 16 and Scellier & Bengio, 2017. We now put this together with Eq. 28 and the finding that $\frac{\partial L}{\partial \beta} = \frac{\mathrm{d}L}{\mathrm{d}\beta}$. Since for the Lipschitz continuous function L in u, W and β (L is even smooth in these arguments), the total derivatives interchange (which is a consequence of the Moore-Osgood theorem applied to the limits of the difference quotients), we then get at any t,

$$\frac{\mathrm{d}C}{\mathrm{d}\boldsymbol{W}} = \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{W}}\frac{\partial L}{\partial \beta} = \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{W}}\frac{\mathrm{d}L}{\mathrm{d}\beta} = \frac{\mathrm{d}}{\mathrm{d}\beta}\frac{\mathrm{d}L}{\mathrm{d}\boldsymbol{W}} = \frac{\mathrm{d}}{\mathrm{d}\beta}\frac{\partial L}{\partial \boldsymbol{W}} = -\frac{\mathrm{d}}{\mathrm{d}\beta}\bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^{\mathrm{T}}\,. \tag{29}$$

The last expression is calculated from the specific form of the Lagrangian (Eq. 17), using that by definition $\bar{\boldsymbol{e}} = \boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}}$.

Finally, in the absence of output nudging, $\beta = 0$, we can assume vanishing errors, $\bar{\boldsymbol{e}} = 0$, as they solve the self-consistency equation, $\bar{\boldsymbol{e}} = \bar{\boldsymbol{r}}'_{\mathrm{net}} \cdot \boldsymbol{W}^{\mathrm{T}}_{\mathrm{net}}\bar{\boldsymbol{e}}$ for all t, see Eq. 27. For these solutions we have $\bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^{\mathrm{T}}\big|_{\beta=0} = 0$. Writing out the total derivative of the function $\boldsymbol{g}(\beta) = \bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^{\mathrm{T}}$ with respect to β at $\beta = 0$ as limit of the difference quotient, $\frac{\mathrm{d}\boldsymbol{g}(\beta)}{\mathrm{d}\beta}\Big|_{\beta=0} = \lim_{\beta\to 0}\frac{1}{\beta}\left(\boldsymbol{g}(\beta) - \boldsymbol{g}(0)\right) = \lim_{\beta\to 0}\frac{1}{\beta}\boldsymbol{g}(\beta)$, using that $\boldsymbol{g}(0) = 0$, we calculate at any t,

$$\frac{\mathrm{d}\,\bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^{\mathrm{T}}}{\mathrm{d}\beta}\bigg|_{\beta=0} = \lim_{\beta\to 0}\frac{1}{\beta}\left(\bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^{\mathrm{T}} - \bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^{\mathrm{T}}\big|_{\beta=0}\right) = \lim_{\beta\to 0}\frac{1}{\beta}\bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^{\mathrm{T}}\,. \tag{30}$$

Here we assume that $\bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^{\mathrm{T}}$ is evaluated at $\beta > 0$ (that itself approaches 0), while $\bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^{\mathrm{T}}\big|_{\beta=0}$ is evaluated at $\beta = 0$. Combining Eq. 29 and 30 yields the cost gradient at any t,

$$-\frac{\mathrm{d}C}{\mathrm{d}\boldsymbol{W}} = \lim_{\beta\to 0}\frac{1}{\beta}\,\bar{\boldsymbol{e}}\,\bar{\boldsymbol{r}}^{\mathrm{T}}\,. \tag{31}$$

This justifies the gradient learning rule $\dot{\boldsymbol{W}}$ in Eq. 27. Learning is stochastic gradient descent on the expected cost, where stochasticity enters in the randomization of the stimulus and target sequences $\boldsymbol{r}_{\mathrm{in}}(t)$ and $\boldsymbol{u}^*(t)$. For the regularity statement, see 'From implicit to explicit differential equations' in the SI. Notice that this proof works for a very general form of the Lagrangian L, until the specific expression for $\frac{\partial L}{\partial \boldsymbol{W}}$. For a proof in terms of partial derivatives only, see SI, where also a primer on partial and total derivatives is found.

Instantaneous gradient descent on $C(\boldsymbol{u}^*_{\boldsymbol{o}}, \bar{\boldsymbol{r}}_{\mathbf{in}})$. The cost $C = \frac{1}{2}\|\boldsymbol{u}^*_{\boldsymbol{o}} - \boldsymbol{u}_{\boldsymbol{o}}\|^2$ at each time t is a function of the voltage $\boldsymbol{u}_{\boldsymbol{o}}$ of the output neurons and the corresponding targets. In a feedforward network, due to the instantaneity of the voltage propagation (Eq. 23), $\boldsymbol{u}_{\boldsymbol{o}}$ is in the absence of output nudging ($\beta = 0$) an instantaneous function of the voltage at the first layer, $\boldsymbol{u}_1(t) = \boldsymbol{W}_{\mathrm{in}}\bar{\boldsymbol{r}}_{\mathrm{in}}(t) + \boldsymbol{u}_1(t_0)\,e^{-\frac{t-t_0}{\tau}}$. For initialisation at $t_0 = -\infty$, the second term vanishes for all t and hence $\boldsymbol{u}_1(t) = \boldsymbol{W}_{\mathrm{in}}\bar{\boldsymbol{r}}_{\mathrm{in}}(t)$. The output voltage $\boldsymbol{u}_{\boldsymbol{o}}(t)$ therefore becomes a function $\boldsymbol{F}_W$ of the low-pass filtered input rate $\bar{\boldsymbol{r}}_{\mathrm{in}}(t)$ that captures the instantaneous network mapping, $\boldsymbol{u}_{\boldsymbol{o}}(t) = \boldsymbol{F}_W(\bar{\boldsymbol{r}}_{\mathrm{in}}(t))$, and with this the cost also becomes an instantaneous function of $\bar{\boldsymbol{r}}_{\mathrm{in}}$ and $\boldsymbol{u}^*_{\boldsymbol{o}}$, namely $C(t) = \frac{1}{2}\|\boldsymbol{u}^*_{\boldsymbol{o}}(t) - \boldsymbol{u}_{\boldsymbol{o}}(t)\|^2 = \frac{1}{2}\|\boldsymbol{u}^*_{\boldsymbol{o}}(t) - \boldsymbol{F}_W(\bar{\boldsymbol{r}}_{\mathrm{in}}(t))\|^2$.

For a general network, again assuming $t_0 = -\infty$, the voltage is determined by the vanishing gradient $\frac{\partial L}{\partial \boldsymbol{u}} = \boldsymbol{f}(\boldsymbol{u}, t) = \boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}}(\boldsymbol{u}) - \bar{\boldsymbol{e}}(\boldsymbol{u}) = 0$ with $\bar{\boldsymbol{e}} = \bar{\boldsymbol{\epsilon}} - \beta\bar{\boldsymbol{e}}^*$, see Eq. 21. For the inclusive treatment of the initial transient see SI, Sects C and D. Remember that $\bar{\boldsymbol{r}} = (\bar{\boldsymbol{r}}_{\mathrm{in}}, \bar{\boldsymbol{r}}_{\mathrm{net}}(\boldsymbol{u}))^{\mathrm{T}}$ and $\bar{\boldsymbol{e}}^* = \boldsymbol{u}^*_{\boldsymbol{o}} - \boldsymbol{u}_{\boldsymbol{o}}$. For a given $\bar{\boldsymbol{r}}_{\mathrm{in}}$ and $\boldsymbol{u}^*_{\boldsymbol{o}}$ at time t, the equation $\boldsymbol{f}(\boldsymbol{u}, t) = 0$ can be locally solved for $\boldsymbol{u}$ if the Hessian $\boldsymbol{H} = \frac{\partial^2 L}{\partial \boldsymbol{u}^2} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}} = \mathbf{1} - \boldsymbol{W}_{\mathrm{net}}\boldsymbol{\rho}' - \bar{\boldsymbol{e}}'$ is invertible, $\boldsymbol{u} = \boldsymbol{F}(\bar{\boldsymbol{r}}_{\mathrm{in}}, \boldsymbol{u}^*_{\boldsymbol{o}})$. This mapping can be restricted to the output voltages $\boldsymbol{u}_{\boldsymbol{o}}$ on the left-hand side, while replacing $\boldsymbol{u}^*_{\boldsymbol{o}} = \boldsymbol{u}_{\boldsymbol{o}} + \bar{\boldsymbol{e}}^*_{\boldsymbol{o}}$ in the argument on the right-hand side (even if this again introduces $\boldsymbol{u}_{\boldsymbol{o}}$ there). With this we obtain the instantaneous mapping $\boldsymbol{u}_{\boldsymbol{o}}(t) = \boldsymbol{F}_W(\bar{\boldsymbol{r}}_{\mathrm{in}}(t), \bar{\boldsymbol{e}}^*_{\boldsymbol{o}}(t))$ from the low-pass filtered input and the output error to the output itself. Notice that for functional feedforward network, the network weight matrix $\boldsymbol{W}_{\mathrm{net}}$ is lower triangular, and for small enough β the Hessian $\boldsymbol{H}$ is therefore always positive definite.

Proving Theorem 2 (rt-DeEL)

Here we restrict ourselves to layered network architectures. To prove Theorem 2 first assume that interneurons receive no nudging ($\beta^{\mathrm{I}} = 0$) and only the lateral interneuron-to-pyramidal weights $\boldsymbol{W}^{\mathrm{PI}}_l$ are plastic. This is already sufficient to prove the rt-DeEL theorem. Yet, simulations showed that shaping the lateral pyramidal-to-interneuron weights $\boldsymbol{W}^{\mathrm{IP}}_l$ so

that it mimics the upper layer activity helps tremendously in learning a correct error representation. We consider this case of learning $\boldsymbol{W}_l^{\mathrm{IP}}$ later.

If the microcircuits is ought to correctly implement error backpropagation, all local prediction errors $\bar{\boldsymbol{e}}_l$ must vanish in the absence of output nudging ($\beta = 0$) as there is no target error. Consequently, any remaining errors in the network are caused by a misalignment of the lateral microcircuit, and we show how learning the interneuron-to-pyramidal weights $\boldsymbol{W}_l^{\mathrm{PI}}$ corrects for such misalignments.

To define the gradient descent plasticity of the weights W_l^{PI} from the interneurons to the pyramidal neurons, we consider the apical error formed by the difference of top-down input and interneuron input, $\bar{\boldsymbol{e}}_l^A = \boldsymbol{B}_l \boldsymbol{u}_{l+1} - \boldsymbol{W}_l^{\mathrm{PI}} \boldsymbol{u}_l^{\mathrm{I}}$, and define the apical mismatch energy as $E_l^{\mathrm{PI}} = \frac{1}{2} \|\bar{\boldsymbol{e}}_l^A\|^2$. Gradient descent along this energy with respect to W_l^{PI} yields

$$\dot{\boldsymbol{W}}_l^{\mathrm{PI}} = \eta^{\mathrm{PI}}\, e_l^A\, \boldsymbol{u}_l^{\mathrm{I}\mathrm{T}} = \eta^{\mathrm{PI}} (\boldsymbol{B}_l \boldsymbol{u}_{l+1} - \boldsymbol{W}_l^{\mathrm{PI}} \boldsymbol{u}_l^{\mathrm{I}})\, \boldsymbol{u}_l^{\mathrm{I}\mathrm{T}}\,, \tag{32}$$

evaluated online while presenting input patterns from the data distribution to the network. We assume that the apical contribution to the somatic voltage is further modulated by the somatic spike rate, $\bar{\boldsymbol{r}}_l' \cdot \bar{\boldsymbol{e}}_l^A$. After successful learning, the top-down input $\boldsymbol{B}_l \boldsymbol{u}_{l+1}$ is fully subtracted away by the lateral input in the apical compartment, and we have

$$\boldsymbol{B}_l \boldsymbol{u}_{l+1} = \boldsymbol{W}_l^{\mathrm{PI}} \boldsymbol{u}_l^{\mathrm{I}}\,. \tag{33}$$

Once this condition is reached, the network achieves a state where, over the activity space spanned by the data, top-down prediction errors throughout the network vanish,

$$\bar{\boldsymbol{e}}_l = \bar{\boldsymbol{r}}_l' \cdot \bar{\boldsymbol{e}}_l^A = \bar{\boldsymbol{r}}_l' \cdot \big(\boldsymbol{B}_l \boldsymbol{u}_{l+1} - \boldsymbol{W}_l^{\mathrm{PI}} \boldsymbol{u}_l^{\mathrm{I}}\big) = 0\,. \tag{34}$$

We show that this top-down prediction error, after the successful learning of the microcircuit, shares the properties of error-backpropagation for a suitable backprojection weights $\boldsymbol{B}$.

Due to the vanishing prediction errors, pyramidal cells only receive bottom-up input $\boldsymbol{u}_{l+1} = \boldsymbol{W}_{l+1} \bar{\boldsymbol{r}}_l$. Using this expression as well as the expression for interneuron membrane potentials without top-down nudging ($\beta^{\mathrm{I}} = 0$ in Eq. 9), $\boldsymbol{u}_l^{\mathrm{I}} = \boldsymbol{W}_l^{\mathrm{IP}} \bar{\boldsymbol{r}}_l$, and plugging both into Eq. 33, we get

$$\boldsymbol{B}_l \boldsymbol{W}_{l+1} \bar{\boldsymbol{r}}_l = \boldsymbol{W}_l^{\mathrm{PI}} \boldsymbol{W}_l^{\mathrm{IP}} \bar{\boldsymbol{r}}_l\,. \tag{35}$$

Assuming that $\boldsymbol{W}_l^{\mathrm{IP}}$ has full rank and the low-pass filtered rates $\bar{\boldsymbol{r}}_l$ span the full n_l dimension of layer l when sampled across the data set we conclude that

$$\boldsymbol{B}_l \boldsymbol{W}_{l+1} = \boldsymbol{W}_l^{\mathrm{PI}} \boldsymbol{W}_l^{\mathrm{IP}}\,. \tag{36}$$

In other words, the loop via upper layer and back is learned to be matched by a lateral loop through the interneurons.

Eq. 36 imposes a restriction on the minimal number of interneurons n_l^{I} at layer l. In fact, the matrix product $\boldsymbol{B}_l \boldsymbol{W}_{l+1}$ maps a n_l-dimensional space onto itself via n_{l+1}-dimensional space. The maximal rank of the this matrix product is limited by the smallest dimension, i.e. $\mathrm{rank}(\boldsymbol{B}_l \boldsymbol{W}_{l+1}) \leq \min(n_l, n_{l+1})$. Analogously, $\mathrm{rank}(\boldsymbol{W}_l^{\mathrm{PI}} \boldsymbol{W}_l^{\mathrm{IP}}) \leq \min(n_l, n_l^{\mathrm{I}})$. But since the two ranks are the same according to Eq. 36, we conclude that in general $n_l^{\mathrm{I}} \geq \min(n_l, n_{l+1})$ must hold, i.e. there should be at least as many interneurons at layer l as the lowest number of pyramidal neurons at either layer l or $l+1$. Note that by choosing $n_l^{\mathrm{I}} = n_{l+1}$ as in (Sacramento *et al.,* 2018) (or $n_l^{\mathrm{I}} > n_{l+1}$ as in this work), the conditions is fulfilled.

With $\boldsymbol{u}_l^{\mathrm{I}} = \boldsymbol{W}_l^{\mathrm{IP}} \bar{\boldsymbol{r}}_l$ and Eq. 36, the top-down prediction error from Eq. 34, in the presence of output nudging ($\beta > 0$), can be written in the backpropagation form

$$\begin{aligned}
\bar{\boldsymbol{e}}_l &= \bar{\boldsymbol{r}}_l' \cdot (\boldsymbol{B}_l\, \boldsymbol{u}_{l+1} - \boldsymbol{W}_l^{\mathrm{PI}}\, \boldsymbol{u}_l^{\mathrm{I}}) = \bar{\boldsymbol{r}}_l' \cdot (\boldsymbol{B}_l\, \boldsymbol{u}_{l+1} - \boldsymbol{W}_l^{\mathrm{PI}}\, \boldsymbol{W}_l^{\mathrm{IP}} \bar{\boldsymbol{r}}_l) && \text{(37a)}\\
&= \bar{\boldsymbol{r}}_l' \cdot (\boldsymbol{B}_l\, \boldsymbol{u}_{l+1} - \boldsymbol{B}_l\, \boldsymbol{W}_{l+1} \bar{\boldsymbol{r}}_l) = \bar{\boldsymbol{r}}_l' \cdot \boldsymbol{B}_l\, (\boldsymbol{u}_{l+1} - \boldsymbol{W}_{l+1} \bar{\boldsymbol{r}}_l) && \text{(37b)}\\
&= \bar{\boldsymbol{r}}_l' \cdot \boldsymbol{B}_l\, \bar{\boldsymbol{e}}_{l+1} = \bar{\boldsymbol{r}}_l' \cdot \boldsymbol{B}_l\, \bar{\boldsymbol{r}}_{l+1}' \cdot \bar{\boldsymbol{e}}_{l+1}^A\,. && \text{(37c)}
\end{aligned}$$

Finally, the simulations showed that learning the lateral weights in the microcircuit greatly benefits from also adapting the pyramidal-to-interneuron weights $\boldsymbol{W}^{\mathrm{IP}}$ by gradient descent on $E^{\mathrm{IP}} = \frac{1}{2} \sum_l \|\boldsymbol{u}_l^{\mathrm{I}} - \boldsymbol{W}_l^{\mathrm{IP}} \bar{\boldsymbol{r}}_l\|^2$, using top-down nudging of the inhibitory neurons ($\beta^{\mathrm{I}} > 0$),

$$\dot{\boldsymbol{W}}_l^{\mathrm{IP}} = \eta^{\mathrm{IP}} (\boldsymbol{u}_l^{\mathrm{I}} - \boldsymbol{W}_l^{\mathrm{IP}} \bar{\boldsymbol{r}}_l) \bar{\boldsymbol{r}}_l^{\mathrm{T}}\,. \tag{38}$$

After learning we have $\boldsymbol{u}_l^{\mathrm{I}} = \boldsymbol{W}_l^{\mathrm{IP}} \bar{\boldsymbol{r}}_l$, and plugging in $\boldsymbol{u}_l^{\mathrm{I}} = (1 - \beta^{\mathrm{I}}) \boldsymbol{W}_l^{\mathrm{IP}} \bar{\boldsymbol{r}}_l + \beta^{\mathrm{I}} \boldsymbol{B}_l^{\mathrm{IP}} \boldsymbol{u}_{l+1}$ (Eq. 9), we obtain $\boldsymbol{W}_l^{\mathrm{IP}} \bar{\boldsymbol{r}}_l = \boldsymbol{B}_l^{\mathrm{IP}} \boldsymbol{u}_{l+1}$. Since $\boldsymbol{u}_{l+1} = \boldsymbol{W}_{l+1} \bar{\boldsymbol{r}}_l$, we conclude as before,

$$\boldsymbol{W}_l^{\mathrm{IP}} = \boldsymbol{B}_l^{\mathrm{IP}}\, \boldsymbol{W}_{l+1}\,. \tag{39}$$

The top-down weights B_l^{IP} that nudge the lower-layer interneurons has randomized entries and may be considered as full rank. If there are less pyramidal neurons in the upper layer than interneurons in the lower layer, B_l^{IP} selects a subspace in the interneuron space of dimension $n_{l+1} < n_l^{\text{I}}$. This seems to simplify the learning of the interneuron-to-pyramidal cell connections W^{PI}. In fact, this learning now has only to match the n_{l+1}-dimensional interneuron subspace embedded in n_l^{I} dimensions to an equal (n_{l+1}-)dimensional pyramidal cell subspace emedded in n_l dimensions.

Learning of the interneuron-to-pyramidal cell connections works with the interneuron nudging as before, and combining Eqs 36 with 39 yields the 'loop consistency'

$$\boldsymbol{B}_l \boldsymbol{W}_{l+1} = \boldsymbol{W}_l^{\text{PI}} \boldsymbol{B}_l^{\text{IP}} \boldsymbol{W}_{l+1} \ . \tag{40}$$

The learning of the microcircuit was described in the absence of output nudging. Conceptually, this is not a problem as one could introduce a pre-learning phase where the lateral connections are first correctly aligned before learning of the feedforward weights begins. In simulations we find that both the lateral connections as well as the forward connections can be trained simultaneously, without the need for such a pre-learning phase. We conjecture that this is due to the fact that our plasticity rules are gradient descent on the energy functions L, E^{PI} and E^{IP} respectively.

Simulation details

The voltage dynamics is solved by a forward-Euler scheme $\boldsymbol{u}(t+dt) = \boldsymbol{u}(t) + \dot{\boldsymbol{u}}(t)\, dt$. The derivative $\dot{\boldsymbol{u}}(t)$ is calculated either through (i) the implicit differential equation (Eq. 7) yielding $\tau \dot{\boldsymbol{u}}(t) = \boldsymbol{h}(\boldsymbol{u}(t), \dot{\boldsymbol{u}}(t-dt))$, or (ii) by isolating $\dot{\boldsymbol{u}}(t)$ and solving for the explicit differential equation $\tau\, \dot{\boldsymbol{u}}(t) = \boldsymbol{g}(\boldsymbol{u}(t))$, as explained after Eq. 49 in the SI.

(i) The implicit differential equation, $\tau \dot{\boldsymbol{u}}(t) = -\boldsymbol{u}(t) + \boldsymbol{W}\boldsymbol{r}(t) + \boldsymbol{e}(t)$, see Eq. 22, is iteratively solved by assigning $\boldsymbol{r}(t) = \rho\big(\boldsymbol{u}(t)\big) + \rho'\big(\boldsymbol{u}(t)\big) \cdot \dot{\boldsymbol{u}}(t-dt)$ and calculating the error $\boldsymbol{e}(t) = \bar{\boldsymbol{e}}(t) + \tau \dot{\bar{\boldsymbol{e}}}(t)$ with $\bar{\boldsymbol{e}}(\boldsymbol{u}) = \rho'(\boldsymbol{u}) \cdot \boldsymbol{W}_{\text{net}}^{\text{T}}\big(\boldsymbol{u} - \boldsymbol{W}_{\text{net}}\rho(\boldsymbol{u}) - \boldsymbol{W}_{\text{in}}\bar{\boldsymbol{r}}_{\text{in}}\big) + \beta \bar{\boldsymbol{e}}^*$ and $\dot{\bar{\boldsymbol{e}}}(t) = \bar{\boldsymbol{e}}'(\boldsymbol{u}(t)) \cdot \dot{\boldsymbol{u}}(t-dt)$.

This iteration exponentially converges to a fixed point $\dot{u}(t)$ on a time scale $\frac{dt}{1-k}$, where $1-k > 0$ is the smallest Eigenvalue of the Hessian $\boldsymbol{H} = \frac{\partial^2 L}{\partial \boldsymbol{u}^2} = \boldsymbol{1} - \boldsymbol{W}_{\text{net}}\boldsymbol{\rho}' - \bar{\boldsymbol{e}}'$, see SI, Sect. C.

(ii) The explicit differential equation is obtained by eliminating the $\dot{\boldsymbol{u}}$ from the right-hand side of the implicit differential equation. Since $\dot{\boldsymbol{u}}$ enters linearly we get $\tau \boldsymbol{H}\, \dot{\boldsymbol{u}} = -\boldsymbol{f} - \tau \frac{\partial \boldsymbol{f}}{\partial t}$ with $\boldsymbol{f}(\boldsymbol{u}, t) = \frac{\partial L}{\partial \boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}} - \bar{\boldsymbol{\epsilon}} - \beta \bar{\boldsymbol{e}}^*$. The explicit form is obtained by matrix inversion, $\dot{\boldsymbol{u}} = \boldsymbol{g}(\boldsymbol{u}, t) = -\frac{1}{\tau}\boldsymbol{H}^{-1}\left(\boldsymbol{f} + \tau \frac{\partial \boldsymbol{f}}{\partial t}\right)$, as the Hessian is invertible if it is strictly positive definite. The external input and the target enter through $\frac{\partial \boldsymbol{f}}{\partial t} = \boldsymbol{W}_{\text{in}} \dot{\bar{\boldsymbol{r}}}_{\text{in}} + \beta\, \dot{\boldsymbol{u}}_{\boldsymbol{o}}^*$, where the derivative of the target voltage is only added for the output neurons $\boldsymbol{o}$. This explicit differential equation is shown to be contractive in the sense that for each input trajectory $\boldsymbol{r}_{\text{in}}(t)$ and target trajectory $\boldsymbol{u}^*(t)$, the voltage trajectory $\boldsymbol{u}(t)$ is locally attracting for neighbouring trajectories. This local attracting trajectory is the vanishing-gradient trajectory $\boldsymbol{f}(\boldsymbol{u}, t) = 0$, and the gradient remains 0 even if the input contains delta-functions, see SI Sect. D.

Solving the explicit differential equation seems to be more robust when the learning rate for $\dot{\boldsymbol{W}}$ gets larger. The explicit form is also less sensitive to large Euler steps dt, see SI Sect. C. By this reason, the ordinary differential equations (ODE) were solved in the explicit form when including plasticity $\dot{\boldsymbol{W}}$. The algorithms are summarized as follows, once without interneurons (Algo 1), and once with interneurons (Algo 2):

Details for Fig. 3b Color coded snapshot of cortical local field potentials (LFPs) in a human brain from 56 deep iEEG electrodes at various locations, converted with the sigmoidal voltage-to-rate function $\bar{r}(u) = \frac{1}{1+e^{-u}}$ and plotted onto a standard Talairach Brain (Talairach & Tournoux, 1988). The iEEG data is from a patient with pharmacoresistant epilepsy and electrodes implanted during presurgical evaluation, extracted from the data release of Burrello *et al.,* 2019. The locations of the electrodes are chosen in accordance with plausibilty, as the original positions of the electrodes were omitted due to ethical standards to prevent patient identification.

Details for Fig. 3c Simulations of the voltage dynamics (Eq. 7a) and weight dynamics (Eq. 8), with learning rate $\eta = 10^{-3}$, step size $dt = 1\,\text{ms}$ for the forward Euler integration, membrane time constant $\tau = 10\,\text{ms}$ and logistic activation function. Weights were initialized randomly from a normal distribution $\mathcal{N}(0, 0.1^2)$ with a cut-off at ± 0.3. The number of neurons in the network $\mathcal{N}$ was $n = 96$, among them 56 output neurons $\mathcal{O} \subset \mathcal{N}$ that were simultaneously nudged, and 40 hidden neurons. During training, all output neurons were nudged simultaneously (with $\beta = 0.1$), whereas during testing, only 42 out of 56 neurons were nudged, the remaining 14 left to reproduce the traces. Data points of the iEEG signal were sampled with a frequency of 512Hz. For simplicity, we therefore assumed that successive data points are separated by 2ms, and up-sampled the signal via simple interpolation to 1ms resolution as required by

Algorithm 1 with projection neurons only, for Figs 3 & 4 (using the explicit ODE, i.e. Step 12 instead of 11)

current state: $\boldsymbol{u}(t)$, $\boldsymbol{W}(t)$
 # consider full vectors and matrices (padded with 0's for feedforward networks)
 # drop time argument (t) for convenience
$\bar{\boldsymbol{r}}_{\text{net}} \leftarrow \rho(\boldsymbol{u})$, $\bar{\boldsymbol{r}} \leftarrow (\bar{\boldsymbol{r}}_{\text{in}}, \bar{\boldsymbol{r}}_{\text{net}})^{\text{T}}$, $\boldsymbol{W} \leftarrow (\boldsymbol{W}_{\text{in}}, \boldsymbol{W}_{\text{net}})$
calculate weight derivatives
$\dot{\boldsymbol{W}} \leftarrow \eta(\boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}})\bar{\boldsymbol{r}}^{\text{T}}$
calculate low-pass-filtered errors
$\bar{\boldsymbol{e}}^*_{\boldsymbol{o}} \leftarrow \boldsymbol{u}^*_{\boldsymbol{o}} - \boldsymbol{u}_{\boldsymbol{o}}$, $\bar{e}^*_i = 0$ for non-output neurons
$\bar{\boldsymbol{e}} \leftarrow \bar{\boldsymbol{r}}'_{\text{net}} \cdot \boldsymbol{W}^{\text{T}}_{\text{net}}(\boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}}) + \beta\bar{\boldsymbol{e}}^*$
calculate temporal voltage derivatives either implicitly (11) **or explicitly** (12)
Implicit: $\tau\dot{\boldsymbol{u}} \leftarrow -\boldsymbol{u} + \boldsymbol{W}(\bar{\boldsymbol{r}} + \tau\dot{\bar{\boldsymbol{r}}}) + (\bar{\boldsymbol{e}} + \tau\dot{\bar{\boldsymbol{e}}})$
Explicit: $\boldsymbol{f} \leftarrow \boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}} - \bar{\boldsymbol{e}}$, $\boldsymbol{H} \leftarrow \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}$, $\dot{\boldsymbol{u}} \leftarrow$ solve $\tau\boldsymbol{H}(\boldsymbol{u})\,\dot{\boldsymbol{u}} = -\boldsymbol{f} - \tau\frac{\partial \boldsymbol{f}}{\partial t}$ via Cholesky decomposition
update voltage and weights
$\boldsymbol{u} \leftarrow \boldsymbol{u} + \dot{\boldsymbol{u}} \cdot dt$, $\boldsymbol{W} \leftarrow \boldsymbol{W} + \dot{\boldsymbol{W}} \cdot dt$

Algorithm 2 including plastic interneurons, for Fig. 5 (using the explicit ODE, i.e. Step 13 instead of 12)

current state: $\boldsymbol{u}(t)$, $\boldsymbol{W}(t)$, $\boldsymbol{u}^{\text{I}}(t)$, $\boldsymbol{W}^{\text{PI}}(t)$, $\boldsymbol{W}^{\text{IP}}(t)$
 # consider full vectors and matrices and drop time argument as in Algorithm 1
$\bar{\boldsymbol{r}} \leftarrow (\bar{\boldsymbol{r}}_{\text{in}}, \rho(\boldsymbol{u}))^{\text{T}}$
calculate weight derivatives
$\dot{\boldsymbol{W}} \leftarrow \eta(\boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}})\bar{r}^{\text{T}}$
$\dot{\boldsymbol{W}}^{\text{PI}} \leftarrow \eta^{\text{PI}}(\boldsymbol{B}\boldsymbol{u} - \boldsymbol{W}^{\text{PI}}\boldsymbol{u}^{\text{I}})\boldsymbol{u}^{\text{I}^{\text{T}}}$
$\dot{\boldsymbol{W}}^{\text{IP}} \leftarrow \eta^{\text{IP}}(\boldsymbol{u}^{\text{I}} - \boldsymbol{W}^{\text{IP}}\bar{\boldsymbol{r}})\bar{\boldsymbol{r}}^{\text{T}}$
calculate low-pass filtered errors
$\bar{\boldsymbol{e}}^*_{\boldsymbol{o}} \leftarrow \boldsymbol{u}^*_{\boldsymbol{o}} - \boldsymbol{u}_{\boldsymbol{o}}$ $(\bar{e}^*_i = 0$ for non-output neurons $i)$
$\bar{\boldsymbol{e}} \leftarrow \boldsymbol{B}\boldsymbol{u} - \boldsymbol{W}^{\text{PI}}\boldsymbol{u}^{\text{I}} + \beta\bar{\boldsymbol{e}}^*$ $(\boldsymbol{B}_{o,:} = \boldsymbol{W}^{\text{PI}}_{o,:} = 0$ for output neurons $o)$
calculate temporal voltage derivatives either implicitly (12) **or explicitly** (13)
Implicit: $\tau\dot{\boldsymbol{u}} \leftarrow -\boldsymbol{u} + \boldsymbol{W}(\bar{\boldsymbol{r}} + \tau\dot{\bar{\boldsymbol{r}}}) + (\bar{\boldsymbol{e}} + \tau\dot{\bar{\boldsymbol{e}}})$
Explicit: $\boldsymbol{f} \leftarrow \boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}} - \bar{\boldsymbol{e}}$, $\boldsymbol{H} \leftarrow \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}$, $\dot{\boldsymbol{u}} \leftarrow$ solve $\tau\boldsymbol{H}(\boldsymbol{u})\,\dot{\boldsymbol{u}} = -\boldsymbol{f} - \tau\frac{\partial \boldsymbol{f}}{\partial t}$ via Cholesky decomposition
update network state
for $\boldsymbol{X} \in \{\boldsymbol{u}, \boldsymbol{W}, \boldsymbol{W}^{\text{PI}}, \boldsymbol{W}^{\text{IP}}\}$ **do**
    $\boldsymbol{X} \leftarrow \boldsymbol{X} + \dot{\boldsymbol{X}}dt$
$\boldsymbol{u}^{\text{I}} \leftarrow (1 - \beta^{\text{I}})\boldsymbol{W}^{\text{IP}}\bar{\boldsymbol{r}} + \beta^{\text{I}}\boldsymbol{B}^{\text{IP}}\boldsymbol{u}$

our integration scheme. Furthermore, the raw values were normalized by dividing them by a factor of 200 to ensure that they are approximately in a range of $\pm 1 - 2$. Training and testing was done on two separate 8s traces of the iEEG recording. Same data as in Fig. 3b1.

Details for Fig. 4 Simulation of the neuronal and synaptic dynamics as given by Eq. 7a, Eq. 7b and Eq. 8. For 5 ms, 10 ms and 50 ms presentation time, we used an integration step size of $dt = 0.05$ ms, $dt = 0.1$ ms and $dt = 0.5$ ms, respectively (and $dt = 1$ ms otherwise). As an activation function, we used the step-linear function (hard sigmoidal) with $\bar{r}(u) = 0$ for $u \leq 0$, $\bar{r}(u) = 1$ for $u \geq 1$ and $\bar{r}(u) = u$ in between. The learning rate was initially set to $\eta = 10^{-3}$ and then reduced to $\eta = 10^{-4}$ after 22 000 s. The nudging strength was $\beta = 0.1$ and the membrane time constant $\tau = 10$ ms. In these simulations (and only for these) we assumed that at each presynaptic layer $l = 0, 1, .., n-1$ there is a first neuron indexed by 0 that fires with constant rate $\bar{r}_{l,0} = 1$, effectively allowing the postsynaptic neurons $\bar{\boldsymbol{r}}_{l+1}$ to learn a bias through the first column of the weight matrix $\boldsymbol{W}_{l+1}$. Weights were initialized randomly from a normal distribution $\mathcal{N}(0, 0.01^2)$ with a cut-off at ± 0.03. For an algorithmic conversion see the scheme below.

Details for Fig. 5 Simulation of neuronal and synaptic dynamics with plastic microcircuit, i.e., the pyramidal-to-interneuron and lateral weights of the microcircuit learned during training.

For the results shown in Fig. 5c2, the following parameters were used. As an activation function, we used a hard sigmoid function and the membrane time constant was set to $\tau = 10$ms. Image presentation time is 100 ms. Forward, pyramidal-to-interneuron and interneuron-to-pyramidal weights were initialized randomly from a normal distribution

$\mathcal{N}(0, 0.01^2)$ with a cut-off at ± 0.03. All learning rates were chosen equal $\eta = 10^{-3}$ and were subsequently reduced to $\eta = 10^{-4}$ after 22000s training time. The nudging parameters were set to $\beta = 0.1$ and $\beta^{\mathrm{I}} = \frac{0.1}{1.1}$. The feedback connections $\boldsymbol{B}_l$ and the nudging matrices $\boldsymbol{B}_l^{\mathrm{IP}}$ were initialized randomly from a normal distribution $5 \cdot \mathcal{N}(0, 0.01^2)$ with a cut-off at ± 0.15. The used integration step size was $dt = 0.25$ms. All weights were trained simultaneously. For an algorithmic conversion see the scheme below. The interneuron membrane potential was calculated by Eq. 9 with a linear transfer function.

Author contributions WS, MAP and DD designed the conceptual approach. WS, DD, MAP, AFK, JJ, JS and YB contributed to different aspects of the framework and model. WS, DD, MAP and AFK derived different components of the theory. DD, BE and AFK performed the simulations. MAP, DD, JJ and BE wrote the first draft of the manuscript. WS wrote the final draft.

References

1. Abbott, L. F., Varela, J. a., Sen, K. & Nelson, S. B. Synaptic depression and cortical gain control. *Science* **275,** 220–224 (1997).
2. Ackley, D. H., Hinton, G. E. & Sejnowski, T. J. A learning algorithm for Boltzmann machines. *Cognitive Science* **9,** 147–169 (1985).
3. Akrout, M., Wilson, C., Humphreys, P. C., Lillicrap, T. & Tweed, D. Deep learning without weight transport. *arXiv* (2019).
4. Bannon, N. M., Chistiakova, M. & Volgushev, M. Synaptic Plasticity in Cortical Inhibitory Neurons: What Mechanisms May Help to Balance Synaptic Weight Changes? *Frontiers in Cellular Neuroscience* **14** (2020).
5. Bartolozzi, C., Indiveri, G. & Donati, E. Embodied neuromorphic intelligence. *Nature Communications* **13,** 1–14 (2022).
6. Bastos, A. M. *et al.* Canonical Microcircuits for Predictive Coding. *Neuron* **76,** 695–711. `http://linkinghub.elsevier.com/retrieve/pii/S0896627312009592` (2012).
7. Bellec, G. *et al.* A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature Communications* **11,** 1–15 (2020).
8. Blom, T., Feuerriegel, D., Johnson, P., Bode, S. & Hogendoorn, H. Predictions drive neural representations of visual events ahead of incoming sensory information. *Proceedings of the National Academy of Sciences of the United States of America* **117,** 7510–7515 (2020).
9. Borovik, A. A mathematician's view of the unreasonable ineffectiveness of mathematics in biology. *Biosystems* **205,** 104410 (2021).
10. Burrello, A., Cavigelli, L., Schindler, K., Benini, L. & Rahimi, A. *Laelaps: An Energy-Efficient Seizure Detection Algorithm from Long-term Human iEEG Recordings without False Alarms* in *Proceedings of the 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)* (2019), 752–757.
11. Coopersmith, J. *The Lazy Universe: An Introduction to the Principle of Least Action* (Oxford University Press, New York, 2017).
12. Dimitriou, M. Enhanced Muscle Afferent Signals during Motor Learning in Humans. *Current Biology* **26,** 1062–1068. `http://dx.doi.org/10.1016/j.cub.2016.02.030` (2016).
13. Dimitriou, M. Human muscle spindles are wired to function as controllable signal-processing devices. *eLife* **11,** 1–14 (2022).
14. Dimitriou, M. & Edin, B. B. Human muscle spindles act as forward sensory models. *Current Biology* **20,** 1763–1767. `http://dx.doi.org/10.1016/j.cub.2010.08.049` (2010).
15. Feldman, A. G. & Levin, M. F. in *Progress in Motor Control* (ed Sternad, D.) 699–726 (2009).
16. Feynman, R., Leighton, R. & Sands, M. *The Feynman Lectures on Physics, Vol. II: Mainly Electromagnetism and Matter, Chapt. 19* `https://www.feynmanlectures.caltech.edu/II_19.html` (Basic Books, 2011).
17. Göltz, J. *et al.* Fast and energy-efficient neuromorphic deep learning with first-spike times. *Nature Machine Intelligence* **3,** 823–835 (2021).
18. Guerguiev, J., Lillicrap, T. P. & Richards, B. A. Towards deep learning with segregated dendrites. *eLife* **6,** 1–37 (2017).
19. Haider, P. *et al.* Latent Equilibrium: Arbitrarily fast computation with arbitrarily slow neurons. *Advances in Neural Information Processing Systems* **34** (2021).
20. Hassabis, D., Kumaran, D., Summerfield, C. & Botvinick, M. Neuroscience-Inspired Artificial Intelligence. *Neuron* **95,** 245–258. `http://dx.doi.org/10.1016/j.neuron.2017.06.011` (2017).

21. Herzfeld, D. J., Vaswani, P. A., Marko, M. K. & Shadmehr, R. A memory of errors in sensorimotor learning. *Science* **345,** 1349–1353. http://www.ncbi.nlm.nih.gov/pubmed/25123484 (2014).
22. Hinton, G. *The Forward-Forward Algorithm : Some Preliminary Investigations* in *NeurIPS* (2022).
23. Hodgkin, A. L. & Huxley, A. F. A quantitative description of membrane current and its application to conduction and excitation in nerve. *Bulletin of Mathematical Biology* **117,** 500–544 (1952).
24. Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proc. Nat. Acad. Sci. USA* **79,** 2554–2558 (1982).
25. Kawato, M. Internal models for motor control and trajectory planning. *Current Opinion in Neurobiology* **9,** 718–727 (1999).
26. Köndgen, H. *et al.* The dynamical response properties of neocortical neurons to temporally modulated noisy inputs in vitro. *Cerebral cortex* **18,** 2086–2097 (2008).
27. Kunin, D. *et al. Two routes to scalable credit assignment without weight symmetry* in *International Conference on Machine Learning* (2020), 5511–5521.
28. Larkum, M. A cellular mechanism for cortical associations: An organizing principle for the cerebral cortex. *Trends in Neurosciences* **36,** 141–151 (2013).
29. Latash, M. L. Muscle coactivation: Definitions, mechanisms, and functions. *Journal of Neurophysiology* **120,** 88–104 (2018).
30. LeCun, Y. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/* (1998).
31. LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M. A. & Huang, F. J. in *Predicting Structured Data* (eds Bakir, G., Hofman, T., Schoelkopf, B., Smola, A. & Taskar, B.) 1–59 (MIT Press, 2006).
32. Li, S. *et al.* Coordinated alpha and gamma control of muscles and spindles in movement and posture. *Frontiers in Computational Neuroscience* **9,** 1–15 (2015).
33. Lillicrap, T. P., Cownden, D., Tweed, D. B. & Akerman, C. J. Random synaptic feedback weights support error backpropagation for deep learning. *Nature communications* **7,** 1–10 (2016).
34. Lillicrap, T. P., Santoro, A., Marris, L., Akerman, C. J. & Hinton, G. Backpropagation and the brain. *Nature Reviews Neuroscience* **21,** 335–346 (2020).
35. Max, K. *et al.* Learning efficient backprojections across cortical hierarchies in real time. *arXiv,* 1–31. http://arxiv.org/abs/2212.10249 (2022).
36. Mesnard, T., Vignoud, G., Sacramento, J., Senn, W. & Bengio, Y. Ghost Units Yield Biologically Plausible Backprop in Deep Neural Networks. *arXiv* (2019).
37. Meulemans, A., Farinha, M. T., *et al.* Credit Assignment in Neural Networks through Deep Feedback Control. *Advances in Neural Information Processing Systems* **34** (2021).
38. Meulemans, A., Zucchet, N., Kobayashi, S., von Oswald, J. & Sacramento, J. The least-control principle for local learning at equilibrium. *Advances in Neural Information Processing Systems* **35** (2022).
39. Ostojic, S. *et al.* Neuronal morphology generates high-frequency firing resonance. *Journal of Neuroscience* **35,** 7056–7068 (2015).
40. Palmer, S. E., Marre, O., Berry, M. J. & Bialek, W. Predictive information in a sensory population. *Proceedings of the National Academy of Sciences* **112,** 6908–6913 (2015).
41. Papaioannou, S. & Dimitriou, M. Goal-dependent tuning of muscle spindle receptors during movement preparation. *Science Advances* **7,** 1–14 (2021).
42. Pfister, J.-P., Dayan, P. & Lengyel, M. Synapses with short-term plasticity are optimal estimators of presynaptic membrane potentials. *Nature Neuroscience* **13,** 1271–1275 (2010).
43. Poirazi, P. & Papoutsi, A. Illuminating dendritic function with computational models. *Nature Reviews Neuroscience* **21,** 303–321 (2020).
44. Rao, R. P. N. & Ballard, D. H. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience* **2,** 79–87. http://www.nature.com/doifinder/10.1038/4580 (1999).
45. Richards, B. A. *et al.* A deep learning framework for neuroscience. *Nature neuroscience* **22,** 1761–1770 (2019).
46. Rumelhart, D. E., Hinton, G. E. & Williams, R. J. Learning Representations by Back-propagating Errors. *Nature* **323,** 533–536 (1986).
47. Sacramento, J., Ponte Costa, R., Bengio, Y. & Senn, W. Dendritic cortical microcircuits approximate the back-propagation algorithm. *Advances in Neural Information Processing Systems* **31,** 8721–8732 (2018).
48. Scellier, B. & Bengio, Y. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in computational neuroscience* **11,** 24 (2017).

49. Schiess, M., Urbanczik, R. & Senn, W. Somato-dendritic Synaptic Plasticity and Error-backpropagation in Active Dendrites. *PLoS Computational Biology* **12,** 1–18 (2016).
50. Simonetto, A., Dall'Anese, E., Paternain, S., Leus, G. & Giannakis, G. B. Time-Varying Convex Optimization: Time-Structured Algorithms and Applications. *Proceedings of the IEEE* **108,** 2032–2048 (2020).
51. Song, Y. *et al.* Inferring Neural Activity Before Plasticity: A Foundation for Learning Beyond Backpropagation. *bioRxiv* (2022).
52. Takahashi, N. *et al.* Active dendritic currents gate descending cortical outputs in perception. *Nature Neuroscience* **23,** 1277–1285. `http://dx.doi.org/10.1038/s41593-020-0677-8` (2020).
53. Talairach, J. & Tournoux, P. *Co-planar stereotaxic atlas of the human brain: 3-Dimensional proportional system: An approach to cerebral imaging* (Thieme Medical Publishers, Inc., New York, 1988).
54. Theis, A. K., Rózsa, B., Katona, G., Schmitz, D. & Johenning, F. W. Voltage gated calcium channel activation by backpropagating action potentials downregulates NMDAR function. *Frontiers in Cellular Neuroscience* **12,** 1–14 (2018).
55. Todorov, E. in *The Bayesian Brain* (eds Doya, K., Ishii, S., Pouget, A. & Rao, R. P.) 12: 1–28 (MIT Press, 2006).
56. Todorov, E. Optimality principles in sensorimotor control. *Nature Neuroscience* **7,** 907–915 (2004).
57. Todorov, E. & Jordan, M. I. Optimal feedback control as a theory of motor coordination. *Nature Neuroscience* **5,** 1226–1235 (2002).
58. Ulrich, D. Dendritic resonance in rat neocortical pyramidal cells. *Journal of Neurophysiology* **87,** 2753–2759 (2002).
59. Urbanczik, R. & Senn, W. Learning by the Dendritic Prediction of Somatic Spiking. *Neuron* **81,** 521–528 (2014).
60. Vogels, T. P., Sprekeler, H., Zenke, F., Clopath, C. & Gerstner, W. Inhibitory Plasticity Balances Excitation and Inhibition in Sensory Pathways and Memory Networks. *Science* **334,** 1569–1573 (2012).
61. Wang, X. D., Chen, C., Zhang, D. & Yao, H. Cumulative latency advance underlies fast visual processing in desynchronized brain state. *Proceedings of the National Academy of Sciences of the United States of America* **111,** 515–520 (2014).
62. Whittington, J. C. & Bogacz, R. An approximation of the error backpropagation algorithm in a predictive coding network with local Hebbian synaptic plasticity. *Neural computation* **29,** 1229–1262 (2017).
63. Whittington, J. C. & Bogacz, R. Theories of error back-propagation in the brain. *Trends in cognitive sciences* **23,** 235–250 (2019).
64. Wigner, E. The Unreasonable Effectiveness of Mathematics in the Natural Sciences. Richard Courant lecture in mathematical sciences. *Communications in pure and applied mathematics* **13,** 1–14 (1959).
65. Wolpert, D. M. & Ghahramani, Z. Computational principles of motor neuroscience. *Nature Neuroscience* **3,** 1212–1217 (2000).
66. Xie, X. & Seung, H. S. Equivalence of backpropagation and contrastive Hebbian learning in a layered network. *Neural computation* **15,** 441–454 (2003).
67. Zenke, F. & Ganguli, S. SuperSpike: Supervised Learning in Multilayer Spiking Neural Networks. *Neural Computation* **30,** 1514–1541 (2018).

Supplementary information

A Extracting the presynaptic voltage error

Threshold-linear transfer functions There is an important special case where the presynaptic voltage error can directly be extracted from presynaptic firing rates, without need to invert the transfer function via synaptic depression as shown below. This is the case when voltage errors in the upper layers are small, and the voltage-to-rate transfer function has derivatives $\rho' = 0$ or 1, so that $\bar{\boldsymbol{r}}'_{l+1} = (\bar{\boldsymbol{r}}'_{l+1})^2$. The condition is satisfied, for instance, for a doubly threshold-linear function (a 'doubly rectified linear unit', ReLu) defined by $\rho(\breve{u}) = 0$ for $\breve{u} < 0$ and $\rho(\breve{u}) = \breve{u}$ for $0 \leq \breve{u} \leq r_{\max}$, while $\rho(\breve{u}) = r_{\max}$ for larger voltages. In this case we calculate

$$\bar{\boldsymbol{e}}_{l+1} = \bar{\boldsymbol{r}}'_{l+1} \cdot \bar{\boldsymbol{e}}^A_{l+1} = (\bar{\boldsymbol{r}}'_{l+1})^2 \cdot \bar{\boldsymbol{e}}^A_{l+1} = \bar{\boldsymbol{r}}'_{l+1} \cdot \bar{\boldsymbol{e}}_{l+1} = \bar{\boldsymbol{r}}'_{l+1} \cdot (\boldsymbol{u}_{l+1} - \boldsymbol{W}_{l+1}\bar{\boldsymbol{r}}_l) \tag{41a}$$

$$\approx \rho(\boldsymbol{u}_{l+1}) - \rho(\boldsymbol{W}_{l+1}\bar{\boldsymbol{r}}_l) = \bar{\boldsymbol{r}}_{l+1} - \rho(\boldsymbol{W}_{l+1}\bar{\boldsymbol{r}}_l)\,. \tag{41b}$$

The approximation uses the Taylor expansion in $\boldsymbol{u}_{l+1}$ and assumes that $\bar{\boldsymbol{e}}_{l+1}$ is small. The crucial point of Eq. 41 is that the mismatch error defined on the voltage, $\bar{\boldsymbol{e}}_{l+1} = \boldsymbol{u}_{l+1} - \boldsymbol{W}_{l+1}\bar{\boldsymbol{r}}_l$, can be factorized into a product of the postsynaptic rate derivative, $\bar{\boldsymbol{r}}'_{l+1}$, and the apical error, and hence it can be expressed as an error defined on the rate. Restricted to

the segment $0 \leq \bar{\boldsymbol{u}}_{l+1} \leq \boldsymbol{r}_{\max}$ where the transfer function is linear and errors do not vanish, the same microcircuit delivers the feedback $\boldsymbol{B}_l \bar{\boldsymbol{r}}_{l+1}$ to the apical tree through the top-down projections, and $-\boldsymbol{B}_l\, \rho(\boldsymbol{W}_{l+1} \bar{\boldsymbol{r}}_l)$ through the lateral connections from the interneurons. While the plasticity rules for $\boldsymbol{W}_l^{\text{PI}}$ and $\boldsymbol{W}_l^{\text{IP}}$ stay the same, the top-down nudging of the interneurons, see Eq. 9, can then be formulated based on the rate instead of the upper layer voltage, $\boldsymbol{u}_l^{\text{I}} = (1-\beta^{\text{I}})\boldsymbol{W}_l^{\text{IP}} \bar{\boldsymbol{r}}_l + \beta^{\text{I}} \boldsymbol{B}_l^{\text{IP}} \bar{\boldsymbol{r}}_{l+1}$, with transfer function of the interneuron again the (doubly) threshold-linear $\rho(\boldsymbol{u}_l^{\text{I}})$. Since voltages and rates are identical in the segment $0 \leq \bar{\boldsymbol{u}}_{l+1} \leq \boldsymbol{r}_{\max}$ for each component, Eqs 36 with 39 can still be inferred.

Rate-to-voltage inversion by short-term synaptic depression We wish to readout the voltage error also for other nonlinear transfer functions than clipped ReLu's. To do so, we take inspiration from the classical short-term synaptic depression model (Tsodyks & Markram, 1997; Abbott *et al.,* 1997; Varela *et al.,* 1997), see also Fig. 6a1. We consider a dynamic vesicle release probability that is proportional to the pool size of available vesicles, $v(\bar{r})$, and this pool size is postulated to depend on past presynaptic rates,

$$p(\text{release} \,|\, \bar{r}) \;\propto\; v(\bar{r}) = 1 + \frac{a}{1 + d\,\bar{r}}\,, \tag{42}$$

where $\bar{r}$ is the low-pass filtered presynaptic rate, a and d are constants. The proportionality factor is $\frac{1}{1+a}$, making a probability out of the vesicle pools size. The effective synaptic strength B of a 'backprojecting top-down' connection is the product of the absolute synaptic strength $B_\circ$ and the vesicle pool size v, i.e. $B = B_\circ\, v(\bar{r})$. The contribution to the postsynaptic current of the synapse is Wr, and the contribution to the postsynaptic voltage is $W\bar{r}$.

We search for an activation function $\bar{r}(u)$ such that the postsynaptic voltage contribution is the scaled presynaptic potential, $B\,\bar{r}(u) = B_\circ u$. Plugging in the above expression for B yields $B\,\bar{r}(u) = B_\circ v(\bar{r})\,\bar{r}(u) = B_\circ u$, and dividing $B_\circ$ out, $v(\bar{r})\,\bar{r}(u) = u$, see Fig. 6a2. With the expression for $v(\bar{r})$ in Eq. 42 we obtain a quadratic equation in $\bar{r}$ that is solved by the non-negative and monotonically increasing function

$$\bar{r} = \frac{1}{2}\left(u - \theta + \sqrt{(u-\theta)^2 + 4u/d}\right) \geq 0\,, \;\text{ for } u \geq 0\,, \tag{43}$$

with 'smooth' threshold $\theta = (1+a)/d$ and asymptote $\bar{r} \approx u - \theta$. This gives us a transfer function $\bar{r}(u)$ that qualitatively matches those observed for pyramidal neurons recorded in the steady state (Anderson *et al.,* 2000; Rauch *et al.,* 2003), see Fig. 6a3.

The approach generalizes to other pairs of strictly monotonic neuronal activation and depression functions $\{\bar{r}(u), v(\bar{r})\}$, as long as u is not driven below some minimal value, here $u_{\text{rest}} = 0$, corresponding to $\bar{r} = 0$. The last requirement can, for instance, be accomplished by offsetting the activation function into a regime that guarantees that u stays positive.

In our simulations for Fig. 5, we did not explicitly implement a dynamic vesicle pool, i.e. the right-hand side of $v(\bar{r})\,\bar{r} = u$, but instead directly used the recovered membrane potentials u.

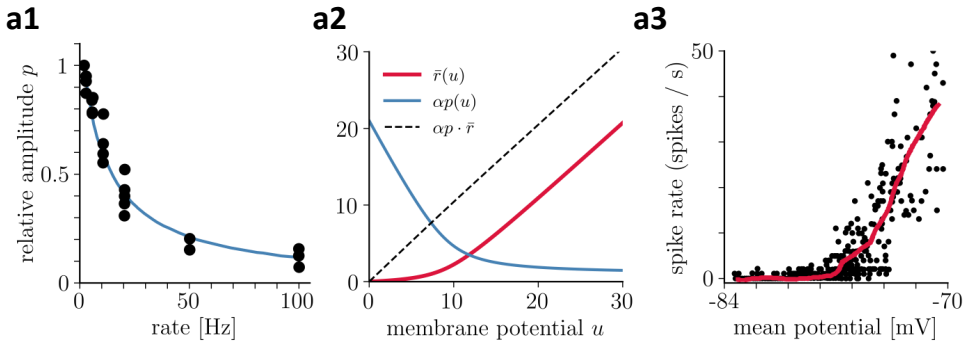

Figure 6: **Recovering presynaptic potentials through short term depression. (a1)** Relative voltage response of a depressing cortical synapse (recreated from Abbott *et al.,* 1997), identified as synaptic release probability p. **(a2)** The product of the low-pass filtered presynaptic firing rate $\bar{r}(u)$ times the synaptic release probability $p(\bar{r})$ is proportional to the presynaptic membrane potential, $p(\bar{r})\,\bar{r} \propto u$. **(a3)** Average in vivo firing rate of a neuron in the visual cortex as a function of the somatic membrane potential (recreated from Anderson *et al.,* 2000), which can be qualitatively identified as the stationary rate $\bar{r}(u)$ derived in Eq. 43.

B Looking back and forward in time with derivatives

Since dealing with extrapolations into the future is a crucial notion of the paper we present here some of the calculations. The discounted future voltage was introduced in Eq. 4 as

$$\tilde{u}(t) = \frac{1}{\tau}\int_t^\infty u(t')e^{(t-t')/\tau}\mathrm{d}t' \,.$$

To show that $\tilde{u}$ satisfies $u = \tilde{u} - \tau\dot{\tilde{u}}$, we need to apply the Leibniz integral rule in calculating the derivative $\dot{\tilde{u}}$. This leads to

$$\frac{\mathrm{d}\tilde{u}}{\mathrm{d}t}(t) = -\frac{1}{\tau}u(t) + \frac{1}{\tau}\int_t^\infty u(t')\frac{1}{\tau}e^{(t-t')/\tau}\mathrm{d}t' \,.$$

Multiplying this equation by τ and using the definition of $\tilde{u}$ yields $\tau\dot{\tilde{u}}(t) = -u(t) + \tilde{u}(t)$, or $u = \tilde{u} - \tau\dot{\tilde{u}}$.

By applying the Leibniz integral rule one also shows that $\bar{x}$, defined in Eq. 15,

$$\bar{x}(t) := \frac{1}{\tau}\int_{-\infty}^t x(t')e^{-\frac{t-t'}{\tau}}\mathrm{d}t' \,,$$

solves $\tau\dot{\bar{x}}(t) = -\bar{x}(t) + x(t)$. This differential equation can be written as $\mathscr{L}\bar{x} = x$, with lookahead operator $\mathscr{L}$ defined in Eq. 14. To show that $\overline{\mathscr{L}x} = \bar{x} + \tau\overline{\dot{x}} = x$, one applies partial integration to $\overline{\dot{x}(t)}$. Note that the equality $\bar{x} + \tau\overline{\dot{x}} = \bar{x} + \tau\dot{\bar{x}} = x$ only holds if we integrate from $-\infty$, and hence if the initialization of the trajectory is far back in the past as compared to the time constant τ.

Uniqueness of the rate function In the main text we concluded from the postulate $r = \rho(u) + \tau\dot{\rho}(u)$ and the general relation $r = \bar{r} + \tau\dot{\bar{r}}$ that $\bar{r}(t) = \rho(u(t))$. This conclusion is a consequence of the uniqueness of a solution of an ordinary differential equation for a given initial condition (that may include delta-functions on the right-hand side, see e.g. Nedeljkov & Oberguggenberger, 2012. In fact, we may consider both variables ρ and $\bar{r}$ as solutions of the differential equation $\tau\dot{x} = -x + r$. Because the solution is unique, we conclude that $\rho = \bar{r}$.

Learning the input time constants In our applications we assumed that the input rates in the original mapping $\boldsymbol{u}_{\boldsymbol{o}}^*(t) = \boldsymbol{F}^*(\bar{\boldsymbol{r}}_{\text{in}}(t))$ are low-pass filtered by a common time constant $\tau^*_{\text{in}(t),i} = \tau$ that is also shared as membrane time constant of the neurons. The general setting of learning to map time series is that input time series, $\boldsymbol{r}_{\text{in}}(t)$ are low-pass filtered with given time constants $\boldsymbol{\tau}^*_{\text{in}}$, and the target output time series $\boldsymbol{u}_{\boldsymbol{o}}^*(t)$ are a function of these low-pass filtered inputs, $\boldsymbol{u}_{\boldsymbol{o}}^*(t) = \boldsymbol{F}_{\boldsymbol{o}}^*(\bar{\boldsymbol{r}}_{\text{in}}^{\boldsymbol{\tau}^*_{\text{in}}}(t))$.

To learn to reproduce the input time constants $\boldsymbol{\tau}^*_{\text{in}}$ in the student network by $\boldsymbol{\tau}_{\text{in}}$, we assume that the inputs converge to neurons $\boldsymbol{u}_1$ that only receive external inputs. Because $\bar{\boldsymbol{r}}_{\text{in}}^{\boldsymbol{\tau}_{\text{in}}} = \boldsymbol{r}_{\text{in}} - \boldsymbol{\tau}_{\text{in}}\cdot\dot{\bar{\boldsymbol{r}}}_{\text{in}}^{\boldsymbol{\tau}_{\text{in}}}$, the gradient rule for the input time constant is

$$\dot{\boldsymbol{\tau}}_{\text{in}} = -\frac{\partial L}{\partial \boldsymbol{\tau}_{\text{in}}} = -\frac{\partial L}{\partial \bar{\boldsymbol{r}}_{\text{in}}^{\boldsymbol{\tau}_{\text{in}}}}^{\mathrm{T}}\frac{\partial \bar{\boldsymbol{r}}_{\text{in}}^{\boldsymbol{\tau}_{\text{in}}}}{\partial \boldsymbol{\tau}_{\text{in}}} = -\boldsymbol{W}_{\text{in}}^{\mathrm{T}}\left(\boldsymbol{u}_1 - \boldsymbol{W}_{\text{in}}\bar{\boldsymbol{r}}_{\text{in}}^{\boldsymbol{\tau}_{\text{in}}}\right)^{\mathrm{T}}\cdot\dot{\bar{\boldsymbol{r}}}_{\text{in}}^{\boldsymbol{\tau}_{\text{in}}} \,. \tag{44}$$

This local learning rule also globally reduces the Lagrangian, and in the limits of $\beta \to \infty$ it is gradient descent on the mismatch energy, while in the limit $\beta \to 0$ it is gradient descent on the cost. The proof works as in Theorem 1. To learn a more complex mapping of time series that includes more complex temporal processing beyond a function of merely $\boldsymbol{u}_{\boldsymbol{o}}^*(t) = \boldsymbol{F}_{\boldsymbol{o}}^*(\bar{\boldsymbol{r}}_{\text{in}}^{\boldsymbol{\tau}^*_{\text{in}}}(t))$, additional variables need to be introduced that form memories (see 'Generalizations:...' below).

C From implicit to explicit differential equations

The original Euler-Lagrange equation is $(1+\tau\frac{\mathrm{d}}{\mathrm{d}t})\frac{\partial L}{\partial \boldsymbol{u}} = 0$, Eq. 20, with the Jacobian $\frac{\partial L}{\partial \boldsymbol{u}} = \boldsymbol{f}(\boldsymbol{u},t) = \boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}} - \bar{\boldsymbol{\epsilon}} - \beta\bar{\boldsymbol{e}}^*$. We have shown that $\mathscr{L}\frac{\partial L}{\partial \boldsymbol{u}} = \boldsymbol{f} + \tau\dot{\boldsymbol{f}} = 0$ is equivalent to the voltage dynamics given in Eq. 22.

Implicit differential equation. The implicit differential equation can be written as

$$\dot{\boldsymbol{u}} = -\frac{\boldsymbol{f}}{\tau} - \frac{\partial \boldsymbol{f}}{\partial t} + \frac{\partial}{\partial \boldsymbol{u}}(\boldsymbol{W}\bar{\boldsymbol{r}} + \bar{\boldsymbol{\epsilon}} + \beta\bar{\boldsymbol{e}}^*)\,\dot{\boldsymbol{u}} = \boldsymbol{K}(\boldsymbol{r}_{\text{in}}, \boldsymbol{u}_{\boldsymbol{o}}^*, \boldsymbol{u}, \dot{\boldsymbol{u}}) \,. \tag{45}$$

The partial derivative of $\boldsymbol{f}(\boldsymbol{u},t)$ with respect to time, $\frac{\partial \boldsymbol{f}}{\partial t}$, captures the external drive from the inputs and output targets that do not depend on $\boldsymbol{u}$, but may directly depend on time. In fact, instead of the argument t of $\boldsymbol{f}$, we could consider the two arguments $\bar{\boldsymbol{r}}_{\text{in}}$ and $\boldsymbol{u}_{\boldsymbol{o}}^*$, and we can then write

$$\frac{\partial f_i}{\partial t} = \frac{\partial f_i}{\partial \bar{\boldsymbol{r}}_{\text{in}}} \dot{\bar{r}}_{\text{in}} + \frac{\partial f_i}{\partial \boldsymbol{u}_{\boldsymbol{o}}^*} \dot{u}_{\boldsymbol{o}}^* = W_{i,\text{in}} \dot{\bar{r}}_{\text{in}} + \beta\, \delta_{io} \dot{\boldsymbol{u}}_{\boldsymbol{o}}^* \,. \tag{46}$$

where δ_{io} is the Kronecker delta and equal to 1 if i is an output neuron, and 0 else. The partial derivatives of $\boldsymbol{f}$ with respect to $\boldsymbol{u}$ represents the (symmetric) Hessian of the Lagrangian, $H_{ij} = \frac{\partial f_i}{\partial u_j} = \frac{\partial^2 L}{\partial u_i \partial u_j} = \delta_{ij} - \frac{\partial}{\partial u_j}(\boldsymbol{W}\bar{\boldsymbol{r}} + \bar{\boldsymbol{\epsilon}} + \beta \bar{\boldsymbol{e}}^*)_i$, with δ_{ij} being the Kronecker-delta, $\bar{\boldsymbol{\epsilon}}$ defined in Eq. 21 and $\bar{\boldsymbol{e}}^*$ defined above Eq. 16. Remember that $\boldsymbol{W} = (\boldsymbol{W}_{\text{in}}, \boldsymbol{W}_{\text{net}})$ and $\bar{\boldsymbol{r}} = (\bar{\boldsymbol{r}}_{\text{in}}, \bar{\boldsymbol{r}}_{\text{net}}) = (\bar{\boldsymbol{r}}_{\text{in}}, \boldsymbol{\rho}(\boldsymbol{u}))$. In vectorial notation the Hessian of the Lagrangian is

$$\boldsymbol{H}(\boldsymbol{u}) = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}} = \frac{\partial^2 L}{\partial \boldsymbol{u}\, \partial \boldsymbol{u}} = \mathbf{1} - \boldsymbol{W}_{\text{net}} \rho'(\boldsymbol{u}) - (\bar{\boldsymbol{\epsilon}} + \beta \bar{\boldsymbol{e}}^*)'(\boldsymbol{u}) \,, \tag{47}$$

where $\mathbf{1}$ is the identity matrix.

Fixing the arguments $(\boldsymbol{r}_{\text{in}}, \boldsymbol{u}_{\boldsymbol{o}}^*, \boldsymbol{u})$ of $\boldsymbol{K}$ in Eq. 45, we need to find a fixed point of the mapping $\dot{\boldsymbol{u}}^{(i+1)} = \boldsymbol{K}(\dot{\boldsymbol{u}}^{(i)})$. In the argument $\dot{\boldsymbol{u}}$, the mapping is affine, $\dot{\boldsymbol{u}}^{(i+1)} = -\frac{\boldsymbol{f}}{\tau} - \frac{\partial \boldsymbol{f}}{\partial t} + \boldsymbol{L}\dot{\boldsymbol{u}}^{(i)}$, with matrix $\boldsymbol{L} = \frac{\partial}{\partial \boldsymbol{u}}(\boldsymbol{W}\bar{\boldsymbol{r}} + \bar{\boldsymbol{\epsilon}} + \beta \bar{\boldsymbol{e}}^*)$. The Banach fixed point theorem asserts that if $\boldsymbol{K}(\dot{\boldsymbol{u}})$ is strictly contracting with k, i.e. if $\|\boldsymbol{K}(\dot{\boldsymbol{u}}^{(2)}) - \boldsymbol{K}(\dot{\boldsymbol{u}}^{(1)})\|^2 \leq k \|\dot{\boldsymbol{u}}^{(2)} - \dot{\boldsymbol{u}}^{(1)}\|$ for all pairs of inputs and $0 \leq k < 1$, then the iteration (here with iteration time step dt) e locally converges to a fixed point $\dot{\boldsymbol{u}} = \boldsymbol{K}(\dot{\boldsymbol{u}})$. Because $\boldsymbol{K}(\dot{\boldsymbol{u}}^{(2)}) - \boldsymbol{K}(\dot{\boldsymbol{u}}^{(1)}) = \boldsymbol{L}(\dot{\boldsymbol{u}}^{(2)} - \dot{\boldsymbol{u}}^{(1)})$, the mapping is k-contractive if the eigenvalues of $\boldsymbol{L}$ have absolute value smaller than k. This is the case if the Hessian $\boldsymbol{H}(\boldsymbol{u}) = \mathbf{1} - \boldsymbol{L}(\boldsymbol{u}) = \mathbf{1} - \boldsymbol{W}\rho'(\boldsymbol{u}) - \bar{\boldsymbol{e}}'(\boldsymbol{u})$, with $\bar{\boldsymbol{e}} = \bar{\boldsymbol{\epsilon}} - \beta \bar{\boldsymbol{e}}^*$, is strictly positive definite. Crucially, because the mapping is affine in $\dot{\boldsymbol{u}}$, the convergence is global.

For strict positive definiteness of the Hession, the Banach fixed point theorem asserts that during (global) convergence the distance to the fixed point is bounded by a constant times $e^{-i\frac{1-k}{dt}}$, with iteration index i and 'virtual' Euler step dt. In an analogue physical device that implements exactly this feedback circuit in continuous time, the dt becomes truly infinitesimal and in this sense the convergence is instantaneous. If dt remains finite, $\dot{\boldsymbol{u}}^{(i)}$ converges to a moving target because the mapping $\dot{\boldsymbol{u}}^{(i+1)} = \boldsymbol{K}(\dot{\boldsymbol{u}}^{(i)})$ changes with time. The target should not change quicker than the time scale $\frac{dt}{1-k}$ of the $\dot{\boldsymbol{u}}^{(i)}$ convergence. Given a time course of the input rate $\boldsymbol{r}_{\text{in}}(t)$ and target $\boldsymbol{u}_o{}^*(t)$ that has bounded variation, the dt can be chosen so that convergence becomes arbitrary quick, and in the limit instantaneous.

If $\boldsymbol{r}_{\text{in}}(t)$ contains well-separated delta-functions, while otherwise still having bounded variations, the reasoning still applies since at any time point in time, except at the time point of the singularity, $\boldsymbol{f}(\boldsymbol{u},t) = 0$. This is shown in Appendix D below.

There is a caveat for the strictly positive definiteness of the Hessian, when the learning rate becomes too big and $\dot{\boldsymbol{W}}$ starts to change the neuronal dynamics. In this case, the Hessian becomes $\boldsymbol{H} = \mathbf{1} - (\boldsymbol{W}\rho' + \dot{\boldsymbol{W}}\rho) - \bar{\boldsymbol{e}}'$, and the Eigenvalues may become negative due to the $\dot{\boldsymbol{W}}$ term. Simulations can in fact become unstable with a big learning rate, and this is more pronounced if also the Euler dt is large. The explicit differential equation avoids the fast iteration towards a moving target and hence allows for larger dt. This in particular pays out in the presence of a high learning rate (although the Cholesky decomposition also requires positive definiteness). By this reason, the large-scale simulations involving plasticity are performed with the explicit form described next.

Explicit differential equation. To isolate $\dot{\boldsymbol{u}}$ in the implicit differential equation, we rewrite $\mathscr{L}\boldsymbol{f} = 0$ again as

$$\tau \dot{\boldsymbol{f}} = \tau \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}} \dot{\boldsymbol{u}} + \frac{\partial \boldsymbol{f}}{\partial t} \right) = -\boldsymbol{f} \,, \quad \text{or} \quad \tau \frac{df_i}{dt} = \tau \left(\sum_j \frac{\partial f_i}{\partial u_j} \dot{u}_j + \frac{\partial f_i}{\partial t} \right) = -f_i \,. \tag{48}$$

Plugging the Hessian $\boldsymbol{H}(\boldsymbol{u}) = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}$ from Eq. 47 into Eq. 48, we obtain the voltage dynamics from Eq. 22 in the equivalent form

$$\tau \boldsymbol{H}(\boldsymbol{u})\, \dot{\boldsymbol{u}} = -\boldsymbol{f} - \tau \frac{\partial \boldsymbol{f}}{\partial t} \,. \tag{49}$$

In our applications, the Hessian $\boldsymbol{H}$ appears to be invertible (although this may not be the case for arbitrary networks), and Eq. 49 can be solved for the unique $\dot{\boldsymbol{u}}$ using the Cholesky decompositions. In fact, if $\boldsymbol{H}$ is invertible, the system implicit ordinary differential equations from Eq. 49 can be converted into a system of explicit ordinary differential equations (with $\boldsymbol{f}(\boldsymbol{u},t) = \frac{\partial L}{\partial \boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}} - \bar{\boldsymbol{\epsilon}} - \beta \bar{\boldsymbol{e}}^*$ and $\boldsymbol{H}$ given in Eq. 47),

$$\dot{\boldsymbol{u}} = \boldsymbol{g}(\boldsymbol{u},t) = -\frac{1}{\tau} \boldsymbol{H}^{-1}(\boldsymbol{u}) \left(\boldsymbol{f}(\boldsymbol{u}) + \tau \frac{\partial \boldsymbol{f}}{\partial t} \right) \,, \tag{50}$$

and as such it has a unique solution for any given initial condition. Because $\frac{\partial f_i}{\partial t} = W_{i,\text{in}}\dot{\bar{\boldsymbol{r}}}_{\text{in}} + \beta\,\delta_{io}\dot{\boldsymbol{u}}_{\boldsymbol{o}}^*$, see Eq. 46, the regularity requirement for $\tau\,\dot{\boldsymbol{u}}$ to be integrable is satisfied even if $\bar{\boldsymbol{r}}_{\text{in}}(t)$ and $\boldsymbol{u}_{\boldsymbol{o}}^*(t)$ contain step functions (and $\boldsymbol{r}_{\text{in}}(t)$ delta-functions), see Sect. D for details.

Even in the presence of such step-functions, the Euler-Lagrange equations $\mathscr{L}\boldsymbol{f} = 0$ lead to an $\boldsymbol{f}$ that is a decaying exponential, $f_i(\boldsymbol{u}(t), t) = c_i\, e^{-\frac{t-t_0}{\tau}}$. For initialization at $t_0 = -\infty$ we have $\boldsymbol{f} = \frac{\partial L}{\partial \boldsymbol{u}} = 0$ at any time. In fact, Eq. 50 is equivalent to $\mathscr{L}\boldsymbol{f} = 0$, and hence any solution of Eq. 50, even if $\frac{\partial \boldsymbol{f}}{\partial t}$ contains a delta-function, is also a solution of $\mathscr{L}\boldsymbol{f} = 0$. Possible jumps in $\bar{\boldsymbol{r}}_{\text{in}}$ or $\boldsymbol{u}_{\boldsymbol{o}}^*$ are compensated by the jumps they induce in $\boldsymbol{u}$ (see below for the full mathematical description with a simple example). To give an intuition, we assume that a recurrent network of our prospevtive neurons has separate fixed points for the two constant input currents $\boldsymbol{r}_{\text{in}}^{(1)}$ and $\boldsymbol{r}_{\text{in}}^{(2)}$. This network still shows an overall relaxation time of $\boldsymbol{\tau}_{\text{in}}$ (but not longer!) when the input rates instantaneously switch from $\boldsymbol{r}_{\text{in}}^{(1)}$ to $\boldsymbol{r}_{\text{in}}^{(2)}$. Nevertheless, at any moment during this relaxation process, gradient learning of the mapping $\boldsymbol{u}_{\boldsymbol{o}}(t) = \boldsymbol{F}_{\boldsymbol{o}}(\bar{\boldsymbol{r}}_{\text{in}}(t))$ towards $\boldsymbol{u}_{\boldsymbol{o}}^*(t) = \boldsymbol{F}_{\boldsymbol{o}}^*(\bar{\boldsymbol{r}}_{\text{in}}(t))$ is still guaranteed (Theorem 1).

In the case of a functional feedforward network, the network weight matrix $\boldsymbol{W}_{\text{net}}$ is lower triangular. This is itself a lower triangular matrix, but now with unit diagonal, and as such it is invertible. For feedforward netowrks, $\boldsymbol{H}$ remains invertible also for small $\beta > 0$, since then the emerging upper diagonal entries remain small compared to the diagonal entries 1. It also remains invertible for growing nudging strengths, $\beta \to \infty$, provided that the weighted target errors $\beta\bar{\boldsymbol{e}}^*$ remain small, see (i) in the proof of Theorem 1.

Link to time-varying optimal control The explicit differential equation, Eq. 50, is a special case of the one in Simonetto, Dall'Anese, *et al.,* 2020, (Eq. 20), where the function to be minimized (their f) can take a general (Lipschitz continuous) form (hence their f is our Lagrangian, $f = L$). To avoid inverting the Hessian, an iteration algorithm can be applied similar to our implicit form form, although more involved to deal with the more general form of L (Simonetto & Dall'Anese, 2017). The idea of tracking the solution of a time-varying optimization problem with a linear look-ahead in time has been introduced introduced in Zhao & Swamy, 1998.

D Contraction analysis and delta-function inputs

Stability We show that the voltage dynamics obtained from $\mathscr{L}\frac{\partial L}{\partial \boldsymbol{u}} = 0$ is always stable, provided that the Hessian $\boldsymbol{H}$ is invertible. For this we rewrite Eq. 49 in the form $\boldsymbol{G}(\boldsymbol{x}, \dot{\boldsymbol{u}}) = \mathscr{L}\frac{\partial L}{\partial \boldsymbol{u}} = \boldsymbol{f} + \tau\dot{\boldsymbol{f}} = 0$, where the explicit time dependence of E and $\boldsymbol{f}$ is a short-cut to express the dependence on $\boldsymbol{x} = (\boldsymbol{r}_{\text{in}}, \dot{\boldsymbol{r}}_{\text{in}}, \boldsymbol{u}_{\boldsymbol{o}}^*, \dot{\boldsymbol{u}}_{\boldsymbol{o}}^*, \boldsymbol{u})^{\text{T}}$. Stated in this generality, the stability analysis also applies to the voltage dynamics derived in the Latent Equilibrium (Haider *et al.,* 2021).

According to the implicit function theorem, at any point in time when $\frac{\partial \boldsymbol{G}}{\partial \dot{\boldsymbol{u}}}$ is invertible, we can locally write $\dot{\boldsymbol{u}} = \boldsymbol{g}(\boldsymbol{x})$. When absorbing the dependence of $\dot{\boldsymbol{u}}$ on $(\boldsymbol{r}_{\text{in}}, \dot{\boldsymbol{r}}_{\text{in}}, \boldsymbol{u}_{\boldsymbol{o}}^*, \dot{\boldsymbol{u}}_{\boldsymbol{o}}^*)$ into a time dependence, we can rewrite this differential equation as $\dot{\boldsymbol{u}} = \boldsymbol{g}(\boldsymbol{u}, t)$, see Eq. 50. This differential equation is contractive and thus stable if the Jacobian of $\boldsymbol{g}$ with respect to $\boldsymbol{u}$ is uniformly negative definite (Lohmiller & Slotine, 1998). The contraction analysis tells that locally, where $\frac{\partial \boldsymbol{G}}{\partial \dot{\boldsymbol{u}}}$ is invertible, we can express $\dot{\boldsymbol{u}}$ as a function $\dot{\boldsymbol{u}} = \boldsymbol{g}(\boldsymbol{x})$ that has derivative $\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}} = -\left(\frac{\partial \boldsymbol{G}}{\partial \dot{\boldsymbol{u}}}\right)^{-1}\frac{\partial \boldsymbol{G}}{\partial \boldsymbol{x}}$. For the $\boldsymbol{u}$-component we get the Jacobian

$$\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}} = -\left(\frac{\partial \boldsymbol{G}}{\partial \dot{\boldsymbol{u}}}\right)^{-1}\frac{\partial \boldsymbol{G}}{\partial \boldsymbol{u}}\,. \tag{51}$$

According to Eq. 49 we have $\boldsymbol{G}(\boldsymbol{x}, \dot{\boldsymbol{u}}) = \boldsymbol{f} + \tau\cdot\boldsymbol{H}(\boldsymbol{u})\,\dot{\boldsymbol{u}} + \tau\frac{\partial \boldsymbol{f}}{\partial t} = 0$ and calculate $\frac{\partial \boldsymbol{G}}{\partial \dot{\boldsymbol{u}}} = \tau\boldsymbol{H}$, with $H_{ij} = \frac{\partial f_i}{\partial u_j}$ specified above. Since according to Eq. 46 the partial derivative with respect to t does not depend on $\boldsymbol{u}$, we also calculate $\frac{\partial \boldsymbol{G}}{\partial \boldsymbol{u}} = \boldsymbol{H} + \tau\frac{\partial \boldsymbol{H}}{\partial \boldsymbol{u}}\dot{\boldsymbol{u}} = \boldsymbol{H} + \tau\frac{d\boldsymbol{H}}{dt}$. Hence, $\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}} = -\frac{1}{\tau}\boldsymbol{H}^{-1}\left(\boldsymbol{H} + \tau\frac{d\boldsymbol{H}}{dt}\right) = -\frac{1}{\tau}\boldsymbol{1} - \frac{d\log \boldsymbol{H}}{dt}$. The term $\frac{d\log \boldsymbol{H}}{dt}$ may cause a violation of the positive definiteness. However, this appears only transiently after initialization since the gradient $\boldsymbol{f}$ exponentially quickly converges to 0 after initialization. If $t_0 = -\infty$, the term $\dot{\boldsymbol{f}}$ vanishes, and with it also $\frac{d\log \boldsymbol{H}}{dt}$. This asserts local contraction property of the fixed point trajectory as then $\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}} = -\frac{1}{\tau}\boldsymbol{1}$. Hence, around a point $\boldsymbol{u}_\circ$ on the trajectory, the linear approximation of the dynamics is $\frac{d}{dt}(\boldsymbol{u} - \boldsymbol{u}_\circ) = \frac{\partial \boldsymbol{g}(\boldsymbol{u}_\circ, t)}{\partial \boldsymbol{u}}(\boldsymbol{u} - \boldsymbol{u}_\circ) = -\frac{1}{\tau}(\boldsymbol{u} - \boldsymbol{u}_\circ)$, showing an exponential local contraction to $\boldsymbol{u}_\circ$.

Global convergence The above stability analysis yields only the strict contraction property after the convergence of the gradient to $\boldsymbol{f} = 0$. We have shown that the iteration of $\boldsymbol{K}(\dot{\boldsymbol{u}})$ globally converges to a fixed point $\dot{\boldsymbol{u}} = \boldsymbol{K}(\dot{\boldsymbol{u}})$, see Eq. 45. Let us therefore assume that $\dot{\boldsymbol{u}}$ satisfies this fixed point equation. This defines a trajectory $\boldsymbol{u}(t)$ that globally converges to the fixed points $\boldsymbol{u}(t) = \boldsymbol{W}\bar{\boldsymbol{r}}(\boldsymbol{u}(t)) + \bar{\boldsymbol{\epsilon}}(\boldsymbol{u}(t)) + \beta\bar{\boldsymbol{e}}^*(\boldsymbol{u}(t))$. In fact, these fixed points are characterized

by the vanishing gradients, $\boldsymbol{f} = 0$, and these vanishing gradients are globally reached. This is because $\boldsymbol{f}(\boldsymbol{u}(t), t)$, as a function of time, exponentially decays to 0 in each component, as noticed above. To restate this in a compound version, we consider an arbitrary initialization $\boldsymbol{u}(t_0)$ for which in general $\boldsymbol{f}(\boldsymbol{u}(t_0), t_0) \neq 0$. Because at any time we have $\mathscr{L}\frac{\partial L}{\partial \boldsymbol{u}} = \boldsymbol{f} + \tau\dot{\boldsymbol{f}} = 0$, the length of $\boldsymbol{f}(\boldsymbol{u}(t), t)$ exponentially converges to 0, as expressed by its temporal derivative,

$$\frac{d}{dt}\frac{1}{2}\|\boldsymbol{f}(\boldsymbol{u}(t), t)\|^2 = \boldsymbol{f}^{\mathrm{T}}\dot{\boldsymbol{f}} = -\boldsymbol{f}^{\mathrm{T}}\tfrac{1}{\tau}\boldsymbol{f} = -\tfrac{1}{\tau}\|\boldsymbol{f}\|^2 < 0\,. \tag{52}$$

Hence, starting at any initial point, the voltage dynamics finds a self-consistency solution of $\boldsymbol{f}(\boldsymbol{u}(t), t) = 0$, or equivalently of $\boldsymbol{u} = \boldsymbol{W}\rho(\boldsymbol{u}) + \bar{\boldsymbol{e}}(\boldsymbol{u})$, with $\bar{\boldsymbol{e}} = \bar{\boldsymbol{\epsilon}} + \beta\bar{\boldsymbol{e}}^*$.

Delta-function inputs keep $\frac{\partial L}{\partial \boldsymbol{u}} = 0$. We next explain in more details why delta-function in the input rates $\boldsymbol{r}_{\text{in}}(t)$ for the NLA the stationarity ('equilibrium') condition $\frac{\partial L}{\partial \boldsymbol{u}} = 0$ is always satisfied (the delta-function in $\boldsymbol{r}_{\text{in}}$ for the NLA corresponding to step-function in $\boldsymbol{r}_{\text{in}}$ for the Latent Equilibrium). We reconsider the explicit differential equation $\dot{\boldsymbol{u}} = \boldsymbol{g}(\boldsymbol{u}, t) = -\frac{1}{\tau}\boldsymbol{H}^{-1}\left(\boldsymbol{f} + \tau\frac{\partial \boldsymbol{f}}{\partial t}\right)$ given in Eq. 50, with $\boldsymbol{f}(\boldsymbol{u}, t) = \frac{\partial L}{\partial \boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}} - \bar{\boldsymbol{\epsilon}} - \beta\bar{\boldsymbol{e}}^*$ and $\boldsymbol{H}$ given in Eq. 47.

To simplify matters, we consider a single delta-function at $t = 0$ as input in the absence of output nudging. In this case we get $\frac{\partial \boldsymbol{f}}{\partial t} = \frac{\partial \boldsymbol{f}}{\partial \bar{\boldsymbol{r}}_{\text{in}}}\dot{\bar{\boldsymbol{r}}}_{\text{in}} + \frac{\partial \boldsymbol{f}}{\partial \bar{\boldsymbol{r}}_{\text{in}}}\dot{\boldsymbol{u}}_{\boldsymbol{o}}^* = \boldsymbol{W}_{\text{in}}\boldsymbol{\delta}_{\text{in}}(t)$, where the input matrix $\boldsymbol{W}_{\text{in}}$ is typically sparse (not all network neurons receive external input), and $\boldsymbol{\delta}_{\text{in}}(t)$ is a vector of delta-functions restricted to the input neurons. Following Nedeljkov & Oberguggenberger, 2012, Proposition 2.1, we can then write the explicit differential equation, Eq. 49, in the form

$$\dot{\boldsymbol{u}} = \boldsymbol{g}(\boldsymbol{u}, t) = \breve{\boldsymbol{g}}(\boldsymbol{u}, t) + \boldsymbol{H}(\boldsymbol{u})^{-1}\boldsymbol{W}_{\text{in}}\boldsymbol{\delta}_{\text{in}}(t)\,, \tag{53}$$

where $\breve{\boldsymbol{g}}(\boldsymbol{u}, t)$ is globally Lipschitz continuous. Due to the Lipschitz continuity the change in $\boldsymbol{u}$ evoked by $\breve{\boldsymbol{g}}(\boldsymbol{u}, t)$ during a small time interval $[-\varepsilon, \varepsilon]$ around $t = 0$ vanishes when this interval shrinks, $\varepsilon \to 0$. To quantifies the change in $\boldsymbol{u}$ during these intervals it is therefore enough to consider $\dot{\boldsymbol{u}} = \boldsymbol{H}(\boldsymbol{u})^{-1}\boldsymbol{W}_{\text{in}}\boldsymbol{\delta}_{\text{in}}(t)$, or equivalently $\boldsymbol{H}(\boldsymbol{u})\dot{\boldsymbol{u}} = \boldsymbol{W}_{\text{in}}\boldsymbol{\delta}_{\text{in}}(t)$. To estimate the jump induced by the delta-functions, we consider some mollifier $\phi_\varepsilon(t) = \varepsilon^{-1}\phi(t/\varepsilon)$, where $\phi(t)$ is a smooth function on the interval $[-1, 1]$ with integral 1. By $\boldsymbol{\phi}_{\text{in},\varepsilon}(t)$ we denote the vector of mollifiers centered at the delta-functions of the input neurons. We now consider the two differential equations, with the second approximating the first on the interval $[-\varepsilon, \varepsilon]$, but without regular term $\breve{\boldsymbol{g}}(\boldsymbol{u}, t)$,

$$\tau\boldsymbol{H}(\boldsymbol{u}_\varepsilon)\,\dot{\boldsymbol{u}}_\varepsilon = \tau\boldsymbol{H}(\boldsymbol{u}_\varepsilon)\breve{\boldsymbol{g}}(\boldsymbol{u}, t) + \boldsymbol{W}_{\text{in}}\boldsymbol{\phi}_{\text{in},\varepsilon}(t) \tag{54a}$$

$$\tau\boldsymbol{H}(\breve{\boldsymbol{u}}_\varepsilon)\,\dot{\breve{\boldsymbol{u}}}_\varepsilon = \boldsymbol{W}_{\text{in}}\boldsymbol{\phi}_{\text{in},\varepsilon}(t)\,, \quad \breve{\boldsymbol{u}}_\varepsilon(-\varepsilon) = \boldsymbol{u}_\varepsilon(-\varepsilon)\,. \tag{54b}$$

We assume that for all $t \in [-\varepsilon, \varepsilon]$ the matrices $\boldsymbol{H}(\boldsymbol{u}_\varepsilon(t))$ and $\boldsymbol{H}(\breve{\boldsymbol{u}}_\varepsilon(t))$ are invertible, so that the two Eqs 54a,b can be turned into an explicit differential equations. Analogously to the 1-dimensional case (Nedeljkov & Oberguggenberger, 2012), we conclude that the solution of Eqs 54a,b on the interval $[-\varepsilon, \varepsilon]$ converge to each other, $\sup_{t\in[-\varepsilon,\varepsilon]}|\boldsymbol{u}_\varepsilon(t) - \breve{\boldsymbol{u}}_\varepsilon(t)| \to 0$ for $\varepsilon \to 0$. As a consequence, the jump of $\breve{\boldsymbol{u}}_\varepsilon$ at $t = 0$ converges to the corresponding jump of $\boldsymbol{u}_\varepsilon$ in the various dimensions.

To calculate the jump of $\breve{\boldsymbol{u}}_\varepsilon$ we have to integrate $\tau\boldsymbol{H}(\breve{\boldsymbol{u}}_\varepsilon)\,\dot{\breve{\boldsymbol{u}}}_\varepsilon$ across the time interval $[-\varepsilon, \varepsilon]$. Instead of integrating $\dot{\breve{\boldsymbol{u}}}_\varepsilon$, we first integrate $\tau\boldsymbol{H}(\breve{\boldsymbol{u}}_\varepsilon)\,\dot{\breve{\boldsymbol{u}}}_\varepsilon$ given in Eq. 54b. Moving from right to left yields

$$\boldsymbol{W}_{\text{in}}\boldsymbol{I}_{\text{in}} = \int_{-\varepsilon}^{\varepsilon}\boldsymbol{W}_{\text{in}}\boldsymbol{\phi}_{\text{in},\varepsilon}(t)\,dt = \tau\int_{-\varepsilon}^{\varepsilon}\boldsymbol{H}(\breve{\boldsymbol{u}}_\varepsilon(t))\,\dot{\boldsymbol{v}}_\varepsilon(t)\,dt = \tau\int_{\breve{\boldsymbol{u}}_\varepsilon(-\varepsilon)}^{\breve{\boldsymbol{u}}_\varepsilon(\varepsilon)}\boldsymbol{H}(\breve{\boldsymbol{u}}_\varepsilon)\;d\breve{\boldsymbol{u}}_\varepsilon\,, \tag{55}$$

where $\boldsymbol{I}_{\text{in}}$ is the index vector of the input neurons having a delta-function, i.e. $I_{\text{in},j} = 1$ if there is a delta-input, and else 0. Note that $\dot{v}_{\varepsilon,j}\,dt = dv_{\varepsilon,j}$. Because $\boldsymbol{H}$ is itself a derivative, $\boldsymbol{H} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}$, we can explicitly calculate the latter integral (also for $\beta > 0$, but for clarity here only done for $\beta = 0$). The last integral in Eq. 55 is defined as a vector with i-th component being

$$\left(\int_{\breve{\boldsymbol{u}}_\varepsilon(-\varepsilon)}^{\breve{\boldsymbol{u}}_\varepsilon(\varepsilon)}\boldsymbol{H}(\breve{\boldsymbol{u}}_\varepsilon)\;d\breve{\boldsymbol{u}}_\varepsilon\right)_i = \sum_{j=1}^{N}\int_{\breve{u}_{\varepsilon,j}(-\varepsilon)}^{\breve{u}_{\varepsilon,j}(\varepsilon)}H_{ij}(u_j)\,du_j = \sum_{j=1}^{N}(\delta_{ij}u_i - W_{ij}\rho(u_j))\big|_{\breve{u}_{\varepsilon,j}(-\varepsilon)}^{\breve{u}_{\varepsilon,j}(\varepsilon)} = \big(\boldsymbol{v}_\varepsilon(\varepsilon) - \boldsymbol{v}_\varepsilon(-\varepsilon)\big)_i\,, \tag{56}$$

where in the second last equality we used that $H_{ij}(\boldsymbol{u}) = \delta_{ij} - W_{ij}\rho'(u_j))$ does only depend on the component u_j, see Eq. 47. In the last equality we introduced the 'network voltage error' $\breve{\boldsymbol{v}}_\varepsilon = \breve{\boldsymbol{u}}_\varepsilon - \boldsymbol{W}_{\text{net}}\rho(\breve{\boldsymbol{u}}_\varepsilon)$. Following the 1-dimensional case treated in Nedeljkov & Oberguggenberger, 2012, Proposition 1.2 we introduce the 'jump function' (called $G(y)$ in the cited work)

$$\boldsymbol{J}(\breve{\boldsymbol{u}}_\varepsilon) = \tau(\breve{\boldsymbol{v}}_\varepsilon - \breve{\boldsymbol{v}}_\circ)\,, \tag{57}$$

with $\check{\boldsymbol{v}}_\circ$ thought to represent $\check{\boldsymbol{v}}_\varepsilon(t_\circ)$ at some time $t_\circ$ before the delta-kick sets in. With this setting, Eq. 55 turns into

$$\boldsymbol{W}_{\text{in}}\boldsymbol{I}_{\text{in}} = \boldsymbol{J}(\check{\boldsymbol{u}}_\varepsilon(\varepsilon)) - \boldsymbol{J}(\check{\boldsymbol{u}}_\varepsilon(-\varepsilon))\,. \tag{58}$$

In the limit $\varepsilon \to 0$ we get a relation between $\boldsymbol{u}(t)$ immediately before and after the jump, $\boldsymbol{u}(-0)$ and $\boldsymbol{u}(+0)$, using that in this limit the boundary points of the trajectories also converge, $\check{\boldsymbol{u}}_\varepsilon(\pm\varepsilon) \to \boldsymbol{u}(\pm 0)$,

$$\boldsymbol{J}(\boldsymbol{u}(+0)) = \boldsymbol{J}(\boldsymbol{u}(-0)) + \boldsymbol{W}_{\text{in}}\boldsymbol{I}_{\text{in}}\,. \tag{59}$$

We now assume that the function $\boldsymbol{J}(\boldsymbol{u}) = \tau(\boldsymbol{v} - \boldsymbol{v}_\circ)$ is invertible around the jump. This is the case if the Jacobian $\frac{\partial \boldsymbol{J}(\boldsymbol{u})}{\partial \boldsymbol{u}}$ is invertible, and because $\boldsymbol{v} = \boldsymbol{u} - \boldsymbol{W}_{\text{net}}\rho(\boldsymbol{u})$, we require invertability of $\frac{\partial \boldsymbol{J}(\boldsymbol{u})}{\partial \boldsymbol{u}} = \tau \boldsymbol{H}(\boldsymbol{u})$, with Hessian defined in Eq. 47.

In the case of invertability we get the voltage after the jump as

$$\boldsymbol{u}(+0) = \boldsymbol{J}^{-1}\big(\boldsymbol{J}(\boldsymbol{u}(-0)) + \boldsymbol{W}_{\text{in}}\boldsymbol{I}_{\text{in}}\big)\,. \tag{60}$$

We next calculate the jump in $\boldsymbol{v}$. This is easy since $\boldsymbol{v} = \boldsymbol{u} - \boldsymbol{W}_{\text{net}}\rho(\boldsymbol{u})$ linearly enters in the function $\boldsymbol{J}(\boldsymbol{u}) = \tau(\boldsymbol{v} - \boldsymbol{v}_\circ)$. Plugging the explicit expression for $\boldsymbol{J}$ into Eq. 59 we get $\tau\,(\boldsymbol{v}(+0) - \boldsymbol{v}_\circ) = \tau\,(\boldsymbol{v}(-0) - \boldsymbol{v}_\circ) + \boldsymbol{W}_{\text{in}}\boldsymbol{I}_{\text{in}}$, or

$$\boldsymbol{v}(+0) = \boldsymbol{v}(-0) + \frac{1}{\tau}\boldsymbol{W}_{\text{in}}\boldsymbol{I}_{\text{in}}\,. \tag{61}$$

Knowing the jump in $\boldsymbol{v}$ helps to show that the equilibrium condition $\frac{\partial L}{\partial \boldsymbol{u}} = 0$ is always satisfied, even immediately after the delta-in put, provided the initialization is at $t_0 = -\infty$. To show this, remember that in the absence of nudging we have $\frac{\partial L}{\partial \boldsymbol{u}} = \boldsymbol{f}(\boldsymbol{u}, t) = \boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}} = \boldsymbol{u} - \boldsymbol{W}_{\text{net}}\rho(\boldsymbol{u}) - \boldsymbol{W}_{\text{in}}\bar{\boldsymbol{r}}_{\text{in}} = \boldsymbol{v} - \boldsymbol{W}_{\text{in}}\bar{\boldsymbol{r}}_{\text{in}}$. The jump size of $\bar{\boldsymbol{r}}_{\text{in}}$ for a delta-function at $t{=}0$, $\boldsymbol{r}_{\text{in}}(t) = \boldsymbol{\delta}_{\text{in}}(t)$ is $\frac{1}{\tau}$. This is because $\bar{\boldsymbol{r}}_{\text{in}}$ satisfies the differential equations $\dot{\bar{\boldsymbol{r}}}_{\text{in}}(t) = -\frac{1}{\tau}\bar{\boldsymbol{r}}_{\text{in}}(t) + \frac{1}{\tau}\boldsymbol{\delta}_{\text{in}}(t)$, provided that $t_0 = -\infty$. Hence,

$$\bar{\boldsymbol{r}}_{\text{in}}(+0) = \bar{\boldsymbol{r}}_{\text{in}}(-0) + \frac{1}{\tau}\boldsymbol{I}_{\text{in}}\,. \tag{62}$$

With Eqs 61 and 62 we conclude that $\frac{\partial L}{\partial \boldsymbol{u}} = \boldsymbol{v} - \boldsymbol{W}_{\text{in}}\bar{\boldsymbol{r}}_{\text{in}} = 0$ throughout.

E Example of a single recurrently connected neuron

To get an intuition for the instantaneity in a the recurrent case we consider the example of a single, recurrently connected neuron. We also put this into the context of the Latent Equilibrium (Haider *et al.,* 2021). Consider the weight vector $\boldsymbol{W} = (W_{\text{in}}, W_{\text{net}})$ with an input an input rate $r_{\text{in}}(t)$ driving the postsynaptic voltage u. The postsynaptic rate is $r = \rho(u) + \tau\dot{\rho}(u)$, and its low-pass filter with respect to τ is $\bar{r} = \rho(u)$. As always, the low-pass filtering reaches back to an initialization at $t_0 = -\infty$, see Eq. 15. The Lagrangian has the form

$$L = \frac{1}{2}\bar{e}^2 + \frac{\beta}{2}\bar{e}^{*^2} = \frac{1}{2}\big(u - (W_{\text{net}}\rho(u) + W_{\text{in}}\bar{r}_{\text{in}})\big)^2 + \frac{\beta}{2}(u^* - u)^2\,. \tag{63}$$

The Euler-Lagrange equations $\mathscr{L}\frac{\partial L}{\partial u} = 0$ are derived from $\frac{\partial L}{\partial u} = \bar{e} - \rho'(u)W_{\text{net}}\bar{\epsilon} - \beta\bar{e}^*$. Applying the look-ahead operator $\mathscr{L}$ (Eq. 14), and abbreviating $\bar{\epsilon} = \rho'(u)W_{\text{net}}\bar{e}$, the Euler-Lagrange equations deliver the voltage dynamics,

$$\tau\,\dot{u} = -u + W_{\text{net}}\big(\rho(u) + \tau\dot{\rho}(u)\big) + W_{\text{in}}r_{\text{in}} + \epsilon + \beta e^*\,. \tag{64}$$

To simplify matters, we consider the nudging-free case, $\beta = 0$. This implies that $\bar{e} = \bar{\epsilon} = 0$. With $\dot{\rho}(u) = \rho'(u)\dot{u}$, we obtain the differential equation

$$\tau\,\left(1 - W_{\text{net}}\rho'(u)\right)\,\dot{u} = -u + W_{\text{net}}\rho(u) + W_{\text{in}}r_{\text{in}}\,. \tag{65}$$

Abbreviating $v = u - W_{\text{net}}\rho(u)$ as 'network voltage error', the above differential equation reads as

$$\tau\,\dot{v} = -v + W_{\text{in}}r_{\text{in}}\,. \tag{66}$$

Integrating the effective voltage dynamics (Eq. 66), assuming initialization infinitely far in the past, is equal to $v = W_{\text{in}}\bar{r}_{\text{in}}$. This equation is equivalent to the Euler-Lagrange equation $\mathscr{L}\frac{\partial L}{\partial u} = 0$ being integrate, and because the solution of the Euler Lagrange equation is $\frac{\partial L}{\partial u} = c\,\boldsymbol{e}^{-\frac{t-t_0}{\tau}}$, we have (using $t_0 = -\infty$)

$$\frac{\partial L}{\partial u} = v - W_{\text{in}}\bar{r}_{\text{in}} = 0\,. \tag{67}$$

Voltage dynamics for a delta-function input We next apply a delta-function in the input rate, say $r_{\text{in}}(t) = \delta(t)$ and consider the dynamics at the level of the voltage, Eq. 65. As in Nedeljkov & Oberguggenberger, 2012, Proposition 1.2 we introduce the 'jump function' (called $G(y)$ in the cited work)

$$J(u) = \tau \int_{u_\circ}^{u} (1 - W_{\text{net}}\rho'(y))\, dy = \tau\Big(\big(u - W_{\text{net}}\rho(u)\big) - v_\circ\Big) = \tau(v - v_\circ)\ . \tag{68}$$

As in Nedeljkov & Oberguggenberger, 2012, Proposition 1.2, we show that the voltage u makes a unique jump at the moment of the delta function that $J(u)$ is invertible around the jump.

We set $v_\circ = u_\circ - W_{\text{net}}\rho(u_\circ)$. Here, $u_\circ$ is some voltage before the jump, say $u_\circ = u(-1)$ evaluated at time $t = -1$, when the jump is at $t = 0$. When $u_0^- = u(-0)$ is the voltage immediately before the jump, the voltage immediately after the jump is $u_0^+ = u(+0)$ specified by

$$J(u_0^+) = J(u_0^-) + W_{\text{in}}\ . \tag{69}$$

The reason is that the W_{in}-scaled delta-function triggers a step of size W_{in} when integrating over it as done in Eq. 68. The new value u_0^+ is unique if J is invertible, and looking at the defining integral in Eq. 68, this is the case if $1 - W_{\text{net}}\rho'(u_0^+) \neq 0$.

The jump in u translates into a jump in $v = u - W_{\text{net}}\rho(u)$ from v_0^- to $v_0^+ = u_0^+ - W_{\text{net}}\rho(u_0^+)$. This endpoint can also be expressed as

$$v_0^+ = v_0^- + \frac{W_{\text{in}}}{\tau}\ . \tag{70}$$

To check this, we assume without loss of generality that $v_\circ = u_\circ - W_{\text{net}}\rho(u_\circ) = 0$. Then $J(u) = \tau(u - W_{\text{net}}\rho(u)) = \tau v$ and $J(u_0^+) = J(u_0^-) + W_{\text{in}} = \tau v_0^- + W_{\text{in}}$ according to Eq. 69. Since also $J(u_0^+) = \tau v_0^+$, we conclude that $\tau v_0^+ = \tau v_0^- + W_{\text{in}}$, as claimed above.

We finally show that even far away from the initialization, the stationarity condition $\frac{\partial L}{\partial u} = 0$ holds before and immediately after the jump. In fact, for $t_0 = -\infty$, the evolution of the 'network voltage error' becomes

$$v(t) = 0 \ \text{ for } \ t \leq 0\,, \quad \text{ and } \quad v(t) = \frac{W_{\text{in}}}{\tau} e^{-\frac{t}{\tau}} \ \text{ for } \ t > 0\ . \tag{71}$$

Here we used that $v_0^- = 0$ and according to Eq. 70 the v jumps to $v_0^- = \frac{W_{\text{in}}}{\tau}$. Remember that for initialization far in the past, $\frac{\partial L}{\partial u} = 0$ is equivalent to $v = W_{\text{in}}\overline{r}_{\text{in}}$, see Eq. 67. We therefore have to calculate the jump in $\overline{r}_{\text{in}}$ induced by the delta-input. Since $\overline{r}_{\text{in}}(t)$ is the solution of $\tau\dot{\overline{r}}_{\text{in}}(t) = -\overline{r}_{\text{in}}(t) + \delta(t)$ for $t_0 = -\infty$, we find that

$$\overline{r}_{\text{in}}(t) = 0 \ \text{ for } \ t \leq 0\,, \quad \text{ and } \quad \overline{r}_{\text{in}}(t) = \frac{1}{\tau} e^{-\frac{t}{\tau}} \ \text{ for } \ t > 0\ . \tag{72}$$

Combing the two Eqs 71 and 72 proves that $\frac{\partial L}{\partial u} = v - W_{\text{in}}\overline{r}_{\text{in}} = 0$ holds true any moment in time, provided the initialization is far in the past.

One may ask why the delta-kink is different from resetting v at a new intialization off from 0. The reason is that at $t = 0$ there is a cause for the jump in $r(0)$, while at t_0 there is no cause in $r(t_0)$. In fact, there is no jump initially, just the start of v at some initial condition. Differently from the initialization at t_0, where $v(t_0) > 0$ implies $\frac{\partial L}{\partial u}(t) > 0$ for finite $t - t_0 > 0$, the jump of $v(0)$ at $t = 0$ to a positive value does leave $\frac{\partial L}{\partial u}(t) = 0$ for all $t > 0$, provided $t_0 = -\infty$.

Linear transfer function We first consider the case of a linear transfer-function $\rho(u) = 0$ (or threshold linear, being in the linear regime). Then the differential equation becomes

$$\tau\,\dot{u} = -u + \frac{W_{\text{in}}}{1 - W_{\text{net}}} r_{\text{in}}\ . \tag{73}$$

With initialization at $t = -\infty$ and low-pass filtering $\overline{r}_{\text{in}}^{\tau}$ defined in Eq. 15 the solution is

$$u(t) = \tfrac{W_{\text{in}}}{1 - W_{\text{net}}}\overline{r}_{\text{in}}^{\tau}(t)\ . \tag{74}$$

The point is that the time constant is τ and is not $\frac{\tau}{1 - W_{\text{net}}}$, as this would be the case without prospective firing rate. In fact, for the 'classical' differential equation,

$$\tau\,\dot{u} = -u + W_{\text{net}}\rho(u) + W_{\text{in}} r_{\text{in}}\ , \tag{75}$$

and ρ the identity, we obtain the differential equation $\tau_{\text{eff}}\,\dot{u} = -u + \frac{W_{\text{in}}}{1 - W_{\text{net}}} r_{\text{in}}$ with $\tau_{\text{eff}} = \frac{\tau}{1 - W_{\text{net}}}$ and solution $u = \frac{W_{\text{in}}}{1 - W_{\text{net}}}\overline{r}_{\text{in}}^{\tau_{\text{eff}}}$ that is now the low-pass filtering with respect to the effective time constant.

Sigmoidal transfer function For a sigmoidal transfer function $\rho(u) = \frac{\bar{r}_{\max}}{1+e^{\vartheta-u}}$, a positive feedback weight $W_{\text{net}} > 0$, and a constant external input r_{in}, say, the solutions $u(t)$ of Eq. 64 either converge to a fixed point or diverge. When converging, the voltage satisfies the fixed point condition

$$u = W_{\text{net}}\rho(u) + W_{\text{in}} r_{\text{in}} \,. \tag{76}$$

This fixed point equation can be numerically solved by time-discrete iteration process. But it can also be solved by a time-continuous process that underlies a neural or neuromorphic implementation. The prospective firing rate introduced in the NLA can be seen as a method to quickly find the fixed point in continuous time. When directly solving the implicit differential equation (as opposed to convert this into an explicit differential equation using e.g. the Cholesky decomposition), the fixed point is potentially found with a fewer number of steps.

To estimate the speed of convergence, we look at the initial speed when taking off at initial condition $u(0)$ between the unstable and stable fixed point. The initial speed for the classical differential equation, Eq. 75, and the NLA version, Eq. 64, are, respectively,

$$\dot{u}(0) = \frac{\Delta u(0)}{\tau} \,, \tag{77a}$$

$$\dot{u}(0) = \frac{\Delta u(0)}{\tau\,(1 - W_{\text{net}}\rho'(u(0)))} \,, \tag{77b}$$

where we set $\Delta u(0) = -u(0) + W_{\text{net}}\rho(u(0)) + W_{\text{in}} r_{\text{in}}(0)$. As $W_{\text{net}}\rho' > 0$, the initial convergence speed of the NLA solution is larger. The scheme has some resemblance to the Newton algorithm of finding zero's of a function by using its derivative.

F Generalizations: NLA for conductance-based neurons and more dynamic variables

The mismatch energies and costs can be generalized in different ways. Here we focus on a biophysical version of the mismatch energy that includes conductance-based neurons. This also relates to the least-action principle in physics. But the NLA can also be generalized to include other dynamica variables such as adaptive thresholds or synaptic short-term plasticity.

Equivalent somato-dentritic circuit For conductance based synapses, the excitatory and inhibitory conductances, g_{E} and g_{I}, are driven by the presynaptic firing rates and have the form $g_{\text{E}}(t) = W_{\text{E}}\, r(t)$, and analogously $g_{\text{I}}(t) = W_{\text{I}}\, r(t)$. The dynamics of a somatic voltage u and a dendritic voltage v reads as

$$c\,\dot{u} \;=\; g_{\text{L}}(E_{\text{L}} - u) + g_{\text{sd}}(v - u) \tag{78}$$

$$c_{\text{d}}\,\dot{v} \;=\; g_{\text{L}}(E_{\text{L}} - v) + g_{\text{E}}(E_{\text{E}} - v) + g_{\text{I}}(E_{\text{I}} - v) + g_{\text{ds}}(u - v) \,, \tag{79}$$

where c and c_{d} are the somatic and dendritic capacitances, $E_{\text{L/E/I}}$ the reversal potentials for the leak, the excitatory and inhibitory currents, g_{sd} the transfer conductance from the dendrite to the soma, and g_{ds} in the other direction.

We consider the case when the dendritic capacitance c_{d} is small as compared to the sum of conductances g_{d} on the right-hand-side of Eq. 79, yielding a fast dendritic time constant. In this case we can solve this equation in the steady state for v, plug this into Eq. 78, and get

$$c\,\dot{u} = g\,(V - u)\,, \tag{80}$$

with effective reversal potential $V = (g_{\text{L}}E_{\text{L}} + g_{\text{sd}}\frac{g_{\text{ff}}}{g_{\text{ff}}+g_{\text{ds}}}v_{\text{ff}})/g$, total conductance $g = g_{\text{L}} + g_{\text{sd}}\frac{g_{\text{ff}}}{g_{\text{ff}}+g_{\text{ds}}}$, feedforward dendritic voltage $v_{\text{ff}} = (g_{\text{L}}E_{\text{L}} + g_{\text{E}}E_{\text{E}} + g_{\text{I}}E_{\text{I}})/g_{\text{ff}}$ and feedforward dendritic conductance $g_{\text{ff}} = g_{\text{L}} + g_{\text{E}} + g_{\text{I}}$. Because $g_{\text{E/I}} = W_{\text{E/I}}\, r(u_{\text{pre}}, \dot{u}_{\text{pre}})$, the conductance depends on the presynaptic voltage and its derivative. Equation 80 describes the effective circuit that has the identical voltage time course as Eqs 78 and 79 with $\dot{v} = 0$, but with a single time-dependent 'battery voltage' V and Ohmic resistance $R = 1/g$.

Somato-dentritic mismatch power and action The synaptic inputs $g_{\text{E}}(t)$ and $g_{\text{I}}(t)$ are continuously driving $V(t)$, and the best what one can hope for the dynamics of u is that it traces V with some integration delay determined by the time constant $\tau = c/g$. In fact, if u follows the dynamics of Eq. 80, then u becomes the low-pass filtered target potential, $u = \overline{V}$, where $V(t)$ is filtered with the dynamic time constant $\tau(t)$. The defining equations for the low-pass filtering is

$$\bar{u} + \tau\dot{\bar{u}} = u \,, \tag{81}$$

and this self-consistency equation is equivalent to the explicit form

$$\bar{u}(t) = \int_{-\infty}^{t} \mathrm{d}t' \, \frac{u(t')}{\tau(t')}\, e^{-\int_{t'}^{t} \mathrm{d}t'' \frac{1}{\tau(t'')}} \,. \tag{82}$$

To capture the voltage dynamics with our NLA principle we recall that the somatic voltage u can be nudged by an 'apical voltage' $\bar{e}$ that causes a somatic voltage error $\bar{e} = u - \overline{V}$. The voltage error drives a current $I = g\bar{e}$ through the conductance g. The electrical power of this current I driven by the voltage $\bar{e}$ is $P = I\bar{e}^2$. This motivates the definition of the mismatch power in a network of N neurons by

$$\mathcal{P} = \sum_i^N \frac{g_i}{2} (u_i - \overline{V}_i)^2 = \frac{1}{2}\boldsymbol{g}^{\mathrm{T}}(\boldsymbol{u} - \overline{\boldsymbol{V}})^2 \,. \tag{83}$$

$\mathcal{P}$ is a virtual power that, nevertheless, is related to some physical flow of ions. Assume we could measure all the ions flowing in the original circuit of Eqs 78 and 79 (in the limit of small ratio $C_\mathrm{d}/g_\mathrm{d}$). From this flow, delete the ion flow that cancels at the level of electrical charge exchange due to the counter directed flow. The remaining effective ion flow defines an effective current flowing through the conductance g with driving force $(V - u)$, Eq. 80. If it were only this effective current in the reduced circuit, the voltage $u(t)$, starting at $u(0)$, would converge with time-constant τ to the low-pass filtering $\overline{V}$. Without additional 'hidden' current, the voltage u would then instantaneously follow $\overline{V}$ that is itself given by the forward dendritic input conductances. The deviation of u from $\overline{V}$, caused by some initial conditions in u or by a feedback current from the network affecting u, builds the mismatch power $\mathcal{P}$. The feedback may originate from a target imposed downstream, and the neuron is 'free' in how to dynamically match u and $\overline{V}$. It is therefore tempting to see $\mathcal{P}$ as a 'free power', and the NLA principle as minimizing the corresponding 'free energy'. In fact, the free-energy principle says that any self-organizing system that is in a dynamic equilibrium with its environment (here $u = \overline{V}$ in the absence of output nudging) must minimize its free energy (that here builds up by imposing a target) (Friston, 2010).

The NLA principle states that the time-integral of $\mathcal{P}$ is minimized with respect to the look-ahead voltage $\tilde{u}$. We therefore define the physical mismatch energy as

$$A = \int_{t_1}^{t_2} \mathcal{P} \,\mathrm{d}t \,, \tag{84}$$

that has the units of energy. $\mathcal{E}_\mathrm{M}$ takes the role of our neural action (A in the main text) for conductance-based neurons.

Euler-Lagrange equations for conductance-based neurons The NLA for conductance-based neurons seeks to minimize $A = \int \mathcal{P}(\boldsymbol{u})\,\mathrm{d}t$ with respect to variations of $\boldsymbol{u}$, such that $\frac{\delta A}{\delta \boldsymbol{u}} = 0$. In the simplest example of the main text we considered prospective rates, $\boldsymbol{r} = \rho(\boldsymbol{u}) + \tau\dot{\rho}(\boldsymbol{u})$, so that the low-pass filtered rates become a function of the instantaneous voltage, $\bar{\boldsymbol{r}} = \rho(\boldsymbol{u})$. These low-pass filtered presynaptic rates, $\bar{\boldsymbol{r}}$, determine the postsynaptic voltage u. Analogously, the low-pass filtered reversal potential, $\overline{\boldsymbol{V}}$, determines the postsynaptic voltage u, and we again postulate that $\overline{\boldsymbol{V}}$ is an instantaneous function of the presynaptic voltage, $\overline{\boldsymbol{V}} = \phi(\boldsymbol{u})$. Here we argue that active dendritic mechanisms advance to postsynaptic reversal potential V, so that the delayed $\overline{V}$ again becomes instantaneous, similarly to the advancement of the apical dendritic potential observed in cortical pyramidal neurons (Ulrich, 2002), see also Fig. 2b. With this instantaneity, the stationarity of the action with respect to generalized (compact and non-compact) variations, $\frac{\delta A}{\delta \boldsymbol{u}} = 0$, translates to the condition $\frac{\partial \mathcal{P}}{\partial \boldsymbol{u}} = 0$.

Calculating $\frac{\partial \mathcal{P}}{\partial \boldsymbol{u}} = 0$ with $\mathcal{P}$ given in Eq. 83 and $\boldsymbol{\tau} = c/\boldsymbol{g}$ for the total conductance $\boldsymbol{g}(\boldsymbol{u})$ specified after Eq. 80 is a bit more demanding. For a probabilistic version, where $\mathcal{P}$ is derived from the negative log-likelihood of a Gaussian density of the voltage, $\mathcal{P} = -\log p(\boldsymbol{u}|\overline{\boldsymbol{V}}) = \frac{1}{2}\boldsymbol{g}^{\mathrm{T}}(\boldsymbol{u} - \overline{\boldsymbol{V}})^2 - \frac{1}{2}\log \boldsymbol{g} + \mathrm{const}$, the calculation is done (Jordan *et al.,* 2022). In the probabilistic version, there is an additional normalization term that enters here as $\log \boldsymbol{g}$. In the deterministic version considered here, this log term is not present and we calculate

$$\frac{\partial \mathcal{P}}{\partial \boldsymbol{u}} = \boldsymbol{g}\cdot\big(\boldsymbol{u} - \overline{\boldsymbol{V}}\big) - \bar{\boldsymbol{\epsilon}} \quad \text{with} \quad \bar{\boldsymbol{\epsilon}} = \bar{\boldsymbol{r}}' \cdot \boldsymbol{W}_\mathrm{net}^{\mathrm{T}} \left((\boldsymbol{u} - \overline{\boldsymbol{V}})\cdot(E^{\mathrm{E/I}} - \overline{\boldsymbol{V}}) - \tfrac{1}{2}(\boldsymbol{u} - \overline{\boldsymbol{V}})^2 \right) \,. \tag{85}$$

Notice that the transpose $\boldsymbol{W}_\mathrm{net}^{\mathrm{T}}$ selects the downstream network neuron to backpropagate from there the first- and second-order errors. From $\frac{\partial \mathcal{P}}{\partial \boldsymbol{u}} = 0$ and $\boldsymbol{\tau} = c/\boldsymbol{g}$ we conclude that

$$\frac{c}{\boldsymbol{\tau}}\cdot\big(\boldsymbol{u} - \overline{\boldsymbol{V}}\big) - \bar{\boldsymbol{\epsilon}} = 0 \quad \text{or} \quad \boldsymbol{u} = \overline{\boldsymbol{V}} + \tfrac{\boldsymbol{\tau}}{c}\cdot\bar{\boldsymbol{\epsilon}} \,. \tag{86}$$

We next apply the look-ahead operator to the expression in this Eq. 86. Assuming an initialization at $t_0 = -\infty$, the condition $\frac{\partial \mathcal{P}}{\partial \boldsymbol{u}} = 0$ becomes equivalent to $\mathscr{L}\frac{\partial \mathcal{P}}{\partial \boldsymbol{u}} = 0$, and hence Eq. 86 becomes equivalent to

$$c\,\dot{\boldsymbol{u}} = \boldsymbol{g}\cdot(\boldsymbol{V} - \boldsymbol{u}) + \boldsymbol{\epsilon} + \dot{\boldsymbol{\tau}}\cdot\bar{\boldsymbol{\epsilon}} \,. \tag{87}$$

A learning rule of the form $\dot{\boldsymbol{W}} \propto \frac{\partial \mathcal{P}}{\partial \boldsymbol{W}} = \bar{\boldsymbol{e}}_\mathrm{post}\,\bar{\boldsymbol{r}}^{\mathrm{T}}$ with an appropriate postsynaptic error $\bar{\boldsymbol{e}}_\mathrm{post}$ can again be derived (see Jordan *et al.,* 2022 for a single neuron in the probabilistic framework). But this, with the above sketch, needs to be worked out yet.

Generalizations: long memories, reinforcement learning One could also extend the NLA principle by adding e.g. threshold adaptation that endows the dynamics with additional and longer time constants. For this, the rate function is parametrized by an additional threshold, $\bar{r} = \rho(u - \vartheta)$, an the Lagrangian is added by an error term on the threshold. Such an error addition can take the form $L = ... + \frac{1}{2}\|\vartheta - \overline{r}^{\tau_\vartheta}\|^2$, where $\overline{r}^{\tau_\vartheta}$ now represents a low-pass filtering of the rate with a long threshold adaptation time constant τ_ϑ. Neurons that show an additional threshold adaptation will still be able to instantaneously transmit a voltage jump through a prospective firing rate, but this will now also depend on the neuron's history. Short-term plasticity may be included in the same way. Due to the history dependence, the stationarity of the action $\delta A = 0$ cannot anymore be reduced to the stationarity of the Lagrangian at any moment in time. As a consequence, the errors will look-ahead into to future more than only based on the local derivatives.

Generalizations are further possible for the cost function that favor voltage regions with high cost, say corresponding to punishment, or negative cost, corresponding to punishment to reward. These extensions will be considered in future work.

Voltage dynamics from the physical least-action principle We have shown that through the look-ahead in Hamilton's least-action principle, the notion of friction enters through the backdoor. In the least action formalism in physics, friction is directly introduced by extending the Hamiltonian principle to a generalized D'Alembert principle, where at the level of the Euler-Lagrange equations the generalized force is equated to the dissipation force (Flannery, 2005).

The electro-chemical properties of a membrane can be captured by an equivalent circuit consisting of a battery voltage V, a conductance C and a resistance R, arranged in parallel. The voltage dynamics is derived from the Euler-Lagrange equations that are added by a dissipative force. Formally, in the absence of an inductance defining the kinetic energy, the Lagrangian $\mathcal{L}$ becomes identical to the potential energy $\mathcal{L} = \mathcal{U}$ with

$$\mathcal{U} = QV - Q^2/(2C)\,, \tag{88}$$

$$\mathcal{F} = R\dot{Q}^2/2\,. \tag{89}$$

Here, $\mathcal{F}$ is the dissipative Rayleigh energy related to friction and Q represents the charge across the membrane. According to D'Alembert's principle, the dynamics is characterized by the Euler-Lagrange equation with dissipative force, $\partial_Q \mathcal{L} - d_t \partial_{\dot{Q}} \mathcal{L} + \partial_{\dot{Q}} \mathcal{F} = 0$, and this equation reduces to $-V + Q/C + R\dot{Q} = 0$. Identifying the charge by means of voltage across the capacitance, $Q = Cu$, this equation can also be written as $-V + u + RC\dot{u} = 0$, or $\tau \dot{u} = V - u$ with $\tau = RC$, just as also derived in Eq. 87. Loosely speaking, the minimization of the energy ($\mathcal{E}_\mathrm{M} = \int \mathcal{P}\,\mathrm{dt}$) by looking ahead is equivalent to a minimization without looking ahead, but taking account of friction.

G A tutorial on total and partial derivatives as used in the paper

The proof of Theorem 1 given in the Methods makes use of partial and total derivatives and follows the notation of Scellier & Bengio, 2017 and Meulemans *et al.,* 2022. As there is some variability (and community-dependent sloppiness) in the notation of partial and total derivatives, we provide some explanations on how this notation is interpreted, and why, for instance, total derivatives commute with each other, and also partial with each other (although not total with partial).

(i) In a differential geometric setting, the derivative of real-valued function $E(\boldsymbol{u})$ on a point $\boldsymbol{u}$ of a manifold (like the flat Euclidean space) is considered as a mapping of tangent vectors at u to the real numbers. When interpreting the derivative as mapping, the $\frac{\partial E}{\partial \boldsymbol{u}}$ is living in the dual space of u and is therefore a row vector if is a column vector. If a function $\boldsymbol{f}(\boldsymbol{u})$ of $\boldsymbol{u}$ is a vector valued, then its derivative is a matrix with entries $\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right)_{ij} = \frac{\partial f_i}{\partial u_j}$, where i is indexing the rows (i.e. running down) and j is indexing the columns (i.e. running right). When $\boldsymbol{r} = \rho(\boldsymbol{u})$ is a column vector with ρ applied to each component of $\boldsymbol{u}$, we consider the (partial) derivative $\boldsymbol{r}' = \rho'(\boldsymbol{u})$ for convenience as column vector with components $\rho'(u_i)$. To strictly follow the formalism, it should be a diagonal matrix.

(ii) Because we introduced the error vector $\bar{\boldsymbol{e}}$ as column vector, it is easier to write $\frac{\partial L}{\partial \boldsymbol{u}} = \bar{\boldsymbol{e}} + ...$, where now $\frac{\partial L}{\partial \boldsymbol{u}}$ is also considered as column vector. To be consistent with the above, we should have written $\frac{\partial L}{\partial \boldsymbol{u}}^\mathrm{T} = \bar{\boldsymbol{e}} + ...$ The $.^\mathrm{T}$ appeared us as too heavy so that we neglected it, where it did not have further mathematical consequences (hoping it does not cause confusions). Sticking here to a column vector also renders the backpropagation error to be a column vector in Eq. 21. This error then gets the classical form with the weight transpose, $\bar{\boldsymbol{\epsilon}} = \bar{\boldsymbol{r}}'_\mathrm{net} \cdot \boldsymbol{W}_\mathrm{net}^\mathrm{T} \bar{\boldsymbol{e}}$.

(iii) We typically have real-valued functions of the form $E(\boldsymbol{u}_{\boldsymbol{\theta}}, \boldsymbol{\theta})$, with $\boldsymbol{\theta} = (\boldsymbol{W}, \beta)$ being a vector of parameters, and $\boldsymbol{u}$ being a function of $\boldsymbol{\theta}$. To get the total derivative of E with respect to $\boldsymbol{\theta}$ we consider the values E

as a function of $\boldsymbol{\theta}$. This can be done by introducing a new function $\mathcal{E}(\boldsymbol{\theta})$ defined as $\mathcal{E}(\boldsymbol{\theta}) = E(\boldsymbol{u}_{\boldsymbol{\theta}}, \boldsymbol{\theta})$, where the components θ_i are considered as independent variables. The total derivative of E with respect to $\boldsymbol{\theta}$ is then defined as vector-valued function $\frac{\mathrm{d}E}{\mathrm{d}\boldsymbol{\theta}} = \left(\frac{\mathrm{d}E}{\mathrm{d}\theta_1}, ..., \frac{\mathrm{d}E}{\mathrm{d}\theta_n}\right) = \left(\frac{\mathrm{d}\mathcal{E}}{\mathrm{d}\theta_1}, ..., \frac{\mathrm{d}\mathcal{E}}{\mathrm{d}\theta_n}\right) = \frac{\partial \mathcal{E}}{\partial \boldsymbol{\theta}}$ for a n-dimensional $\boldsymbol{\theta}$. It can be helpful to think of the components of this total derivative as a (total) directional derivatives in the unit directions. For the last (n-th) unit direction $\Delta\boldsymbol{\theta}^{(n)} = (0, .., 0, 1)$, for instance, we have $\frac{\mathrm{d}E}{\mathrm{d}\theta_n} = \lim_{\varepsilon \to 0} \frac{1}{\varepsilon} \left(E(\boldsymbol{u}_{\boldsymbol{\theta}+\varepsilon\Delta\boldsymbol{\theta}^{(n)}}, \boldsymbol{\theta} + \varepsilon\Delta\boldsymbol{\theta}^{(n)}) - E(\boldsymbol{u}_{\boldsymbol{\theta}}, \boldsymbol{\theta})\right)$.

(iv) The cost gradient, $\frac{\mathrm{d}C}{\mathrm{d}\boldsymbol{W}} \propto (\boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}})\,\bar{\boldsymbol{r}}^T$ has the same dimension as $\boldsymbol{W}$. Recall that by the cost gradient we mean $\frac{\mathrm{d}C}{\mathrm{d}\boldsymbol{W}} = \frac{\partial \mathcal{C}}{\partial \boldsymbol{W}}$, where $\mathcal{C}$ is defined as $\mathcal{C}(\boldsymbol{W}) = C(\boldsymbol{u}_{\mathbf{o}}(\boldsymbol{W}), \boldsymbol{u}^*_{\mathbf{o}})$, with the voltage $\boldsymbol{u}_{\mathbf{o}}$ of the output neurons being itself a function of $\boldsymbol{W}$.

(v) To calculate the partial derivative $\frac{\partial}{\partial \boldsymbol{\theta}} E(\boldsymbol{u}, \boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$, we fix the first argument $\boldsymbol{u}_{\boldsymbol{\theta}}$, even if for $\boldsymbol{u}$ we often plugged in the components of the trajectory $\boldsymbol{u} = \boldsymbol{u}_{\boldsymbol{\theta}}(t)$ that now does depend on $\boldsymbol{\theta}$. In contrast, the total derivative is $\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}} E(\boldsymbol{u}, \boldsymbol{\theta}) = \frac{\partial E(\boldsymbol{u},\boldsymbol{\theta})}{\partial \boldsymbol{u}} \frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}\boldsymbol{\theta}} + \frac{\partial E(\boldsymbol{u},\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$. Here, $\frac{\partial E(\boldsymbol{u},\boldsymbol{\theta})}{\partial \boldsymbol{u}}$ is a row vector, as also $\frac{\partial E(\boldsymbol{u},\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$, consistent with the convention (that we have broken in Eq. 28 to keep vectors as columns). When $\boldsymbol{u}$ is considered as trajectory, $\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}\boldsymbol{\theta}}$ does not vanish in general, but it does when $\boldsymbol{u}$ is simply considered as independent variable.

(vi) When replacing the argument $\boldsymbol{u}$ in $E(\boldsymbol{u}, \boldsymbol{\theta})$ by $\boldsymbol{u} = \tilde{\boldsymbol{u}} - \tau\dot{\tilde{\boldsymbol{u}}}$ we get the 'Lagrangian' $L(\tilde{\boldsymbol{u}}, \dot{\tilde{\boldsymbol{u}}}, \boldsymbol{\theta}) = E(\tilde{\boldsymbol{u}} - \tau\dot{\tilde{\boldsymbol{u}}}, \theta)$. The partial derivative of L with respect to $\tilde{\boldsymbol{u}}$, for instance, is then $\frac{\partial}{\partial \tilde{\boldsymbol{u}}} L(\tilde{\boldsymbol{u}}, \dot{\tilde{\boldsymbol{u}}}, \boldsymbol{\theta}) = \frac{\partial}{\partial \boldsymbol{u}} E(\boldsymbol{u}, \boldsymbol{\theta}) \frac{\partial \boldsymbol{u}}{\partial \tilde{\boldsymbol{u}}} = \frac{\partial}{\partial \boldsymbol{u}} E(\boldsymbol{u}, \boldsymbol{\theta})$. The partial derivative $\frac{\partial}{\partial \tilde{\boldsymbol{u}}} L$ considers L as a function of independent arguments $\tilde{\boldsymbol{u}}$, $\dot{\tilde{\boldsymbol{u}}}$ and $\boldsymbol{\theta}$.

(vii) We also used that the total derivatives can be exchanged, in the current example $\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{W}} \frac{\mathrm{d}}{\mathrm{d}\beta} E = \frac{\mathrm{d}}{\mathrm{d}\beta} \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{W}} E$. This is generally true for derivatives of Lipschitz continuous functions, for which derivatives exist almost everywhere. The total derivatives (where they exist) then commute because the difference quotients in $\boldsymbol{W}$ and β are uniformly bounded. The Moore-Osgood theorem tells that two limits, of which at least one is uniform in the other, can be commuted. This also applies to the double difference quotients involved in the definition of $\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{W}} \frac{\mathrm{d}}{\mathrm{d}\beta}$. Remember that the total derivative, for instance with respect to the n-th parameter, can be written as $\frac{\mathrm{d}E}{\mathrm{d}\theta_n} = \lim_{\varepsilon \to 0} \frac{1}{\varepsilon} \left(E(\boldsymbol{u}_{\boldsymbol{\theta}+\varepsilon\Delta\boldsymbol{\theta}^{(n)}}, \boldsymbol{\theta} + \varepsilon\Delta\boldsymbol{\theta}^{(n)}) - E(\boldsymbol{u}_{\boldsymbol{\theta}}, \boldsymbol{\theta})\right)$.

H Proof of Theorem 1, part (ii), using only partial derivatives

This Section proves Eqs 29–31 in terms with only partial derivatives, banning nested functions that require to deal with total derivatives. The 3 equations are also the core for the proof for Equilibrium Propagation Scellier & Bengio, 2017, although there only applied in the steady state after the network converged to a constant activity.

We assume a network of d neurons whose membrane potential is given by $\boldsymbol{u} \in \mathbb{R}^d$ and which are connected via weights $\boldsymbol{W} \in \mathbb{R}^{d\times d}$. By $\boldsymbol{\nabla}_{\boldsymbol{u}}$, we denote the gradient with respect to the membrane potentials, i.e., $\boldsymbol{\nabla}_{\boldsymbol{u}} = (\frac{\partial}{\partial u_1}, \dots, \frac{\partial}{\partial u_d})$. Similarly, $\boldsymbol{\nabla}_{\boldsymbol{W}}$ is a matrix containing the derivatives with respect to the weights, $(\boldsymbol{\nabla}_{\boldsymbol{W}})_{ij} = \frac{\partial}{\partial W_{ij}}$.

To prove the rt-DeEP theorem (theorem 1), we first have to make a few definitions and observations:

1. For a given $(\boldsymbol{W}, \beta)$, the dynamics yield certain membrane potentials $f_{\boldsymbol{u}} \in \mathbb{R}^d$. Formally, we define this as

$$f_{\boldsymbol{u}} : \mathbb{R}^{d\times d} \times \mathbb{R} \to \mathbb{R}^d\,, \quad (\boldsymbol{W}, \beta) \mapsto f_{\boldsymbol{u}}(\boldsymbol{W}, \beta)\,. \tag{90}$$

   The ith element of $f_{\boldsymbol{u}}$ is denoted by f_{u_i} and hence $\boldsymbol{\nabla}_{f_{\boldsymbol{u}}} = (\frac{\partial}{\partial f_{u_1}}, \dots, \frac{\partial}{\partial f_{u_d}})$.

2. We define the following functions:

   - the mismatch energy $E^{\mathrm{M}} : \mathbb{R}^d \times \mathbb{R}^{d\times d} \to \mathbb{R}\,,\ (\boldsymbol{u}, \boldsymbol{W}) \mapsto E^{\mathrm{M}}(\boldsymbol{u}, \boldsymbol{W}) = \frac{1}{2} \sum_{i=1}^d \left[u_i - \sum_j W_{ij} \bar{r}_j\right]^2$,
   - the cost function $C : \mathbb{R}^d \to \mathbb{R}\,,\ \boldsymbol{u} \mapsto C(\boldsymbol{u}) = \frac{1}{2} \sum_{k\in\mathcal{O}} \|u^*_k - u_k\|^2$,
   - the Lagrangian $L : \mathbb{R}^d \times \mathbb{R}^{d\times d} \times \mathbb{R} \to \mathbb{R}\,,\ (\boldsymbol{u}, \boldsymbol{W}, \beta) \mapsto L(\boldsymbol{u}, \boldsymbol{W}, \beta) = E^{\mathrm{M}}(\boldsymbol{u}, \boldsymbol{W}) + \beta C(\boldsymbol{u})$.

   To make the dependency of the cost and energies on β and $\boldsymbol{W}$ explicit, we further introduce three auxiliary functions F_{M}, F_C and F_L:

   - for the mismatch energy $F_{\mathrm{M}} : \mathbb{R}^{d\times d} \times \mathbb{R} \to \mathbb{R}\,,\ (\boldsymbol{W}, \beta) \mapsto F_{\mathrm{M}}(\boldsymbol{W}, \beta) = E^{\mathrm{M}}\left(f_{\boldsymbol{u}}(\boldsymbol{W}, \beta), \boldsymbol{W}\right)$,
   - for the cost function $F_C : \mathbb{R}^{d\times d} \times \mathbb{R} \to \mathbb{R}\,,\ (\boldsymbol{W}, \beta) \mapsto F_C(\boldsymbol{W}, \beta) = C\left(f_{\boldsymbol{u}}(\boldsymbol{W}, \beta)\right)$,
   - for the Lagrangian $F_L : \mathbb{R}^{d\times d} \times \mathbb{R} \to \mathbb{R}\,,\ (\boldsymbol{W}, \beta) \mapsto F_L(\boldsymbol{W}, \beta) = F_{\mathrm{M}}(\boldsymbol{W}, \beta) + \beta F_C(\boldsymbol{W}, \beta)$

3. The Euler-Lagrange equations can be written as $\tau \frac{\mathrm{d}}{\mathrm{d}t} \boldsymbol{\nabla}_{\boldsymbol{u}} L = -\boldsymbol{\nabla}_{\boldsymbol{u}} L$. Hence, far enough away from initialization and for smooth enough input (and targets) we have $\boldsymbol{\nabla}_{f_{\boldsymbol{u}}} L = 0$ at all times, even when changing the network input continuously. Note that both the cost and the mismatch energies are defined on low-pass-filtered signals, and it is with respect to the low-pass filtered external input that the low-pass-filtered output error is minimized.

4. Without output nudging (i.e., $\beta = 0$), the output error vanishes and consequently all other prediction errors vanish as well, $\bar{\boldsymbol{e}} = \boldsymbol{u} - \boldsymbol{W}\bar{\boldsymbol{r}} = 0$. This can be easily shown for layered network architectures and holds true for arbitrary connections (e.g., recurrent networks) as long as $f_{\boldsymbol{u}}$ uniquely exists, i.e., $\boldsymbol{u} = \boldsymbol{W}\bar{\boldsymbol{r}}$ has a unique solution for $\boldsymbol{u}$. From the form of the mismatch energy, we then get

$$\boldsymbol{\nabla}_{\boldsymbol{W}} E^{\mathrm{M}}\big|_{\beta=0} = \left(\boldsymbol{W}\,\bar{\boldsymbol{r}} - \boldsymbol{u}\right)\bar{\boldsymbol{r}}^{\mathrm{T}}\big|_{\beta=0} = 0\,. \tag{91}$$

   Since we are assuming smooth functions, this also implies that

$$\lim_{\epsilon_\beta \to 0} \boldsymbol{\nabla}_{\boldsymbol{W}} E^{\mathrm{M}}\big|_{\beta=\epsilon_\beta} = 0\,. \tag{92}$$

5. From the assumption of well-behaved (smooth) functions, it also follows that partial derivatives commute $\boldsymbol{\nabla}_{\boldsymbol{W}} \frac{\partial}{\partial\beta} = \frac{\partial}{\partial\beta}\boldsymbol{\nabla}_{\boldsymbol{W}}$.

Our goal is to find a plasticity rule that minimizes the cost C, which we do by calculating $\boldsymbol{\nabla}_{\boldsymbol{W}} F_C\big|_{\beta=0}$. Similar to Scellier & Bengio (2017), to achieve this, we first calculate the partial derivatives of F_L with respect to the nudging strength β

$$\begin{aligned}
\frac{\partial F_L}{\partial\beta} &= \frac{\partial F_{\mathrm{M}}}{\partial\beta} + \beta\frac{\partial F_C}{\partial\beta} + F_C && \text{(93a)}\\
&= \sum_{i=1}^{d}\left(\frac{\partial E^{\mathrm{M}}}{\partial f_{u_i}}\frac{\partial f_{u_i}}{\partial\beta} + \beta\frac{\partial C}{\partial f_{u_i}}\frac{\partial f_{u_i}}{\partial\beta}\right) + C && \text{(93b)}\\
&= \sum_{i=1}^{d}\underbrace{\frac{\partial\left(E^{\mathrm{M}} + \beta C\right)}{\partial f_{u_i}}}_{=0}\frac{\partial f_{u_i}}{\partial\beta} + C && \text{(93c)}\\
&= C\,, && \text{(93d)}
\end{aligned}$$

and the weights $\boldsymbol{W}$

$$\begin{aligned}
\frac{\partial F_L}{\partial W_{ij}} &= \frac{\partial F_{\mathrm{M}}}{\partial W_{ij}} + \beta\frac{\partial F_C}{\partial W_{ij}} && \text{(94a)}\\
&= \sum_{k=1}^{d}\left(\frac{\partial E^{\mathrm{M}}}{\partial f_{u_k}}\frac{\partial f_{u_k}}{\partial W_{ij}} + \beta\frac{\partial C}{\partial f_{u_k}}\frac{\partial f_{u_k}}{\partial W_{ij}}\right) + \frac{\partial E^{\mathrm{M}}}{\partial W_{ij}} && \text{(94b)}\\
&= \sum_{k=1}^{d}\underbrace{\frac{\partial\left(E^{\mathrm{M}} + \beta C\right)}{\partial f_{u_k}}}_{=0}\frac{\partial f_{u_k}}{\partial W_{ij}} + \frac{\partial E^{\mathrm{M}}}{\partial W_{ij}} && \text{(94c)}\\
&= \frac{\partial E^{\mathrm{M}}}{\partial W_{ij}}\,, && \text{(94d)}\\
& && \text{(94e)}
\end{aligned}$$

or in vectorized form

$$\boldsymbol{\nabla}_{\boldsymbol{W}} F_L = \boldsymbol{\nabla}_{\boldsymbol{W}} E^{\mathrm{M}}\,. \tag{95}$$

With these identities in place, we can calculate the plasticity rule:

$$
\begin{aligned}
-\lim_{\epsilon_\beta \to 0} \boldsymbol{\nabla}_{\boldsymbol{W}} F_C &= \lim_{\epsilon_\beta \to 0} \boldsymbol{\nabla}_{\boldsymbol{W}} \frac{\partial F_L}{\partial \beta} && \text{(96a)}\\
&= \lim_{\epsilon_\beta \to 0} \frac{\partial}{\partial \beta} \boldsymbol{\nabla}_{\boldsymbol{W}} F_L && \text{(96b)}\\
&= \lim_{\epsilon_\beta \to 0} \lim_{\delta \to 0} \frac{1}{\delta} \left(\boldsymbol{\nabla}_{\boldsymbol{W}} F_L \big|_{\beta=\epsilon_\beta+\delta} - \boldsymbol{\nabla}_{\boldsymbol{W}} F_L \big|_{\beta=\epsilon_\beta} \right) && \text{(96c)}\\
&= \lim_{\delta \to 0} \frac{1}{\delta} \lim_{\epsilon_\beta \to 0} \left(\boldsymbol{\nabla}_{\boldsymbol{W}} E^{\mathrm{M}} \big|_{\beta=\epsilon_\beta+\delta} - \boldsymbol{\nabla}_{\boldsymbol{W}} E^{\mathrm{M}} \big|_{\beta=\epsilon_\beta} \right) && \text{(96d)}\\
&= \lim_{\delta \to 0} \frac{1}{\delta} \left(\lim_{\epsilon_\beta \to 0} \boldsymbol{\nabla}_{\boldsymbol{W}} E^{\mathrm{M}} \big|_{\beta=\epsilon_\beta+\delta} - \underbrace{\lim_{\epsilon_\beta \to 0} \boldsymbol{\nabla}_{\boldsymbol{W}} E^{\mathrm{M}} \big|_{\beta=\epsilon_\beta}}_{=0 \text{ from Eq. 92}} \right) && \text{(96e)}\\
&= \lim_{\delta \to 0} \frac{1}{\delta} \boldsymbol{\nabla}_{\boldsymbol{W}} E^{\mathrm{M}} \big|_{\beta=\delta} && \text{(96f)}\\
&\approx \frac{1}{\delta} \boldsymbol{\nabla}_{\boldsymbol{W}} E^{\mathrm{M}} \big|_{\beta=\delta} \quad \text{for } \delta \ll 1\,, && \text{(96g)}
\end{aligned}
$$

where we used that limits can be exchanged for smooth functions. Using the definition of E^{M}, we obtain a plasticity rule that minimizes the cost function

$$
-\lim_{\epsilon_\beta \to 0} \boldsymbol{\nabla}_{\boldsymbol{W}} C \big|_{\beta=\epsilon_\beta} \approx -\frac{1}{\epsilon_\beta} \boldsymbol{\nabla}_{\boldsymbol{W}} E^{\mathrm{M}} \big|_{\beta=\epsilon_\beta} = \frac{1}{\epsilon_\beta} \left(\boldsymbol{u} - \boldsymbol{W}\, \bar{\boldsymbol{r}} \right) \bar{\boldsymbol{r}}^{\mathrm{T}} \big|_{\beta=\epsilon_\beta} \quad \text{for } \epsilon_\beta \ll 1\,. \tag{97}
$$

In practice, the prefactor $\frac{1}{\epsilon_\beta}$ is absorbed into the learning rate.

References

1. Abbott, L. F., Varela, J. a., Sen, K. & Nelson, S. B. Synaptic depression and cortical gain control. *Science* **275,** 220–224 (1997).
2. Anderson, J. S., Lampl, I., Gillespie, D. C. & Ferster, D. The contribution of noise to contrast invariance of orientation tuning in cat visual cortex. *Science* **290,** 1968–1972 (2000).
3. Flannery, M. R. The enigma of nonholonomic constraints. *American Journal of Physics* **73,** 265–272 (2005).
4. Friston, K. The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience* **11,** 127–138 (2010).
5. Haider, P. *et al.* Latent Equilibrium: Arbitrarily fast computation with arbitrarily slow neurons. *Advances in Neural Information Processing Systems* **34** (2021).
6. Jordan, J., Sacramento, J., Wybo, W. A. M., Petrovici, M. A. & Senn, W. Learning Bayes-optimal dendritic opinion pooling. *arXiv*. `http://arxiv.org/abs/2104.13238` (2022).
7. Lohmiller, W. & Slotine, J.-j. E. On Contraction Analysis for Non-linear Systems. *Automatica* **34,** 683–696 (1998).
8. Meulemans, A., Zucchet, N., Kobayashi, S., von Oswald, J. & Sacramento, J. The least-control principle for local learning at equilibrium. *Advances in Neural Information Processing Systems* **35** (2022).
9. Nedeljkov, M. & Oberguggenberger, M. Ordinary differential equations with delta function terms. *Publications de l'Institut Mathematique* **91,** 125–135 (2012).
10. Rauch, A., La Camera, G., Lüscher, H.-R., Senn, W. & Fusi, S. Neocortical Pyramidal Cells Respond as Integrate-and-Fire Neurons to In Vivo-Like Input Currents. *Journal of Neurophysiology* **90,** 1598–1612 (2003).
11. Scellier, B. & Bengio, Y. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in computational neuroscience* **11,** 24 (2017).
12. Simonetto, A. & Dall'Anese, E. Prediction-Correction Algorithms for Time-Varying Constrained Optimization. *IEEE Transactions on Signal Processing* **65,** 5481–5494 (2017).
13. Simonetto, A., Dall'Anese, E., Paternain, S., Leus, G. & Giannakis, G. B. Time-Varying Convex Optimization: Time-Structured Algorithms and Applications. *Proceedings of the IEEE* **108,** 2032–2048 (2020).
14. Tsodyks, M. & Markram, H. The neural code between neocortical pyramidal neurons depends on neurotransmitter release probability. *Proc. Nat. Acad. Sci. USA* **94,** 719–723 (1997).

15. Ulrich, D. Dendritic resonance in rat neocortical pyramidal cells. *Journal of Neurophysiology* **87,** 2753–2759 (2002).
16. Varela, J. A. *et al.* A quantitative description of short-term plasticity at excitatory synapses in layer 2/3 of rat primary visual cortex. *Journal of Neuroscience* **17,** 7926–7940 (1997).
17. Zhao, Y. & Swamy, M. N. Novel technique for tracking time-varying minimum and its applications. *Canadian Conference on Electrical and Computer Engineering* **2,** 910–913 (1998).