Beyond accuracy: generalization properties of bio-plausible temporal credit assignment rules

Anonymous Author(s)
Affiliation
Address
email

Abstract

To unveil how the brain learns, ongoing work seeks biologically-plausible approximations of gradient descent algorithms for training recurrent neural networks (RNNs). Yet, beyond task accuracy, it is unclear if such learning rules converge to solutions that exhibit different levels of generalization than their non-biologically-plausible counterparts. Leveraging results from deep learning theory based on loss landscape curvature, we ask: how do biologically-plausible gradient approximations affect generalization? We first demonstrate that state-of-the-art biologically-plausible learning rules for training RNNs exhibit worse and more variable generalization performance compared to their machine learning counterparts that follow the true gradient more closely. Next, we verify that such generalization performance is correlated significantly with loss landscape curvature, and we show that biologically-plausible learning rules tend to approach high-curvature regions in synaptic weight space. Using tools from dynamical systems, we derive theoretical arguments and present a theorem explaining this phenomenon. This predicts our numerical results, and explains why biologically-plausible rules lead to worse and more variable generalization properties. Finally, we suggest potential remedies that could be used by the brain to mitigate this effect. To our knowledge, our analysis is the first to identify the reason for this generalization gap between artificial and biologically-plausible learning rules, which can help guide future investigations into how the brain learns solutions that generalize.

1 Introduction

A longstanding question in neuroscience is how animals excel at learning complex behavior involving temporal dependencies across multiple timescales and thereafter generalize this learned behavior to unseen data. This requires solving the temporal credit assignment problem: how to assign the contribution of past neural states to future outcomes. To address this, neuroscientists are increasingly turning to the mathematical framework provided by recurrent neural networks (RNNs) to model learning mechanisms in the brain [1, 2]. Temporal credit assignment in RNNs is typically achieved by backpropagation through time (BPTT), or other gradient descent-based optimization algorithms, none of which are biologically-plausible (or bio-plausible for short). Therefore, the use of RNNs as a framework to understand the computational principles of learning in the brain has motivated an influx of bio-plausible learning rules that approximate gradient descent [1–3].

Performance of such rules is typically quantified by accuracy. Although the accuracy achieved by these rules is often comparable to true gradient descent, little is known about the breadth of the emergent solutions, namely how well they generalize. Broadly speaking, generalization refers to a trained model's ability to adapt to previously unseen data, and is typically measured by the so-called *generalization gap*: the difference between training and testing error. This is especially

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

important when learning complex tasks with nonlinear RNNs where the loss landscape is non-convex, and therefore, many solutions with comparable training accuracy can exist. These solutions, characterized as (local) minima in the loss landscape, can nonetheless exhibit drastically different levels of generalization (Figure 1). It is not clear if gradient-based methods like BPTT and the existing bio-plausible alternatives have a different tendency to converge to loss minima that provide better or worse generalization.

While better predictors of the generalization performance remains an open research question in deep learning [4], recent extensive studies identify flatness of the loss landscape at the solution point as a promising predictor for generalization [5–9]. Leveraging these empirical and theoretical findings, **we ask**: how do proposed biologically-motivated gradient approximations affect flatness of the converged solution on loss landscape, and thereby, generalization?

Our overarching goal is to investigate generalization trends for existing bio-plausible temporal credit assignment rules, and in particular, examine how truncation-based bio-plausible gradient approximations can affect such trends. Specifically, **our contributions** are summarized as follows:

- In numerical experiments, we demonstrate across several well-known neuroscience and machine learning benchmark tasks that state-of-the-art (SoTA) bio-plausible learning rules for training RNNs exhibit *worse* and *more variable* generalization gaps, compared to true (stochastic) gradient descent (Figure 2A-C).
- Using the same experiments, we show that bio-plausible learning rules tend to approach high-curvature regions in synaptic weight space as measured by the loss' Hessian eigenspectrum (Figure 2D-F). Further, we verify that this correlates with worse generalization (Figure 2D-F and 3), which is consistent with the literature.
- We present a theorem to explain this phenomenon by examining the weight update equation as a discrete dynamical system, which sheds light on a potential connection between gradient alignment and the preference over converging to narrower minima (Theorem 1, Figure 4).

Given the core components in designing artificial neural networks: data, objective functions, learning rules and architectures [10], we investigate different learning rules while holding the data, objective function and architecture constant. SoTA RNN learning rules investigated include a three-factor rule with symmetric feedback (symmetric e-prop [11]), a three-factor rule using random feedback weights (RFLO [12]) and a multi-factor rule using local modulatory signaling (MDGL) [13]. We also discuss potential remedies used by the brain (Appendix Figure 5) To our knowledge, our analysis is the first to highlight and quantitatively provide a mechanistic explanation of the reason for this gap in solution quality between artificial and bio-plausible learning rules for RNNs, thereby motivating further investigations into how the brain learns solutions that generalize.

2 Related works

2.1 Bio-plausible gradient approximations

Investigating bio-plausible learning rules is of interest both to identify more efficient training strategies for artificial networks and to better understand learning in the brain [2, 3, 14–43]. In order for learning to reach a certain goal quantified by an objective, learning algorithms often minimize a loss function [10]. The error gradient, if available, tells us how each parameter should be adjusted in order to have the steepest local descent. For training RNNs, which are widely used as a model for neural circuits [44–63], standard algorithms that follow this gradient — real time recurrent learning (RTRL) and BPTT — are not bio-plausible and have overwhelming memory storage demands [3, 64]. However, learning rules that only approximate the true gradient can sometimes be as effective as those that follow the gradient exactly [10, 65]. Because of that, bio-plausible learning rules that approximate the gradient using known properties of real neurons have been proposed and led to successful learning outcomes across many different tasks in feedforward networks [1, 66–75], with recent extensions to recurrently connected networks [11–13]. These existing bio-plausible rules for training RNNs [11–13] are truncation-based (which is the focus of this study), so that the untruncated terms of the gradient can be assigned with putative identities to known biological learning ingredients: eligibility traces, which maintain preceding activity on the molecular levels [76–81], combined with top-down instructive signaling [76, 77, 82–88] as well as local cell-to-cell

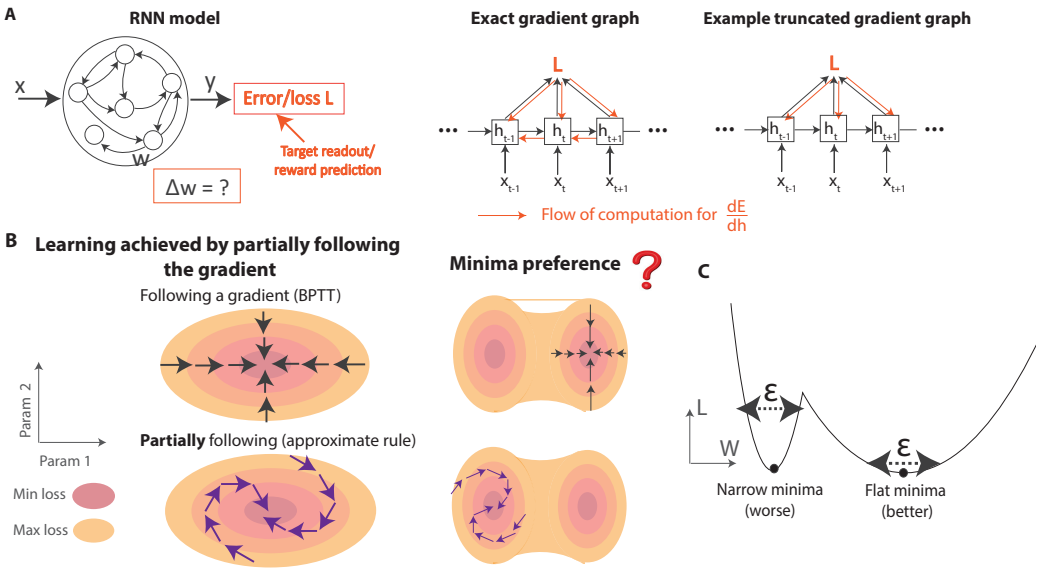

Figure 1: **Setup**. A) Illustration of an RNN trained to minimize error/loss function L (left). Existing bio-plausible proposals for RNNs estimate the gradient by neglecting dependencies that are biologically implausible to compute (right). B) Low training error/loss can be achieved by *partially* following a gradient (right), but preference for converging to minima with certain generalization properties remains underexamined for these learning rules (left). C) Minima flatness matters: 1-D loss landscape illustration with two solutions that equally minimize loss L, but exhibit drastically different generalization properties: the narrower minima is more sensitive to perturbation.

modulatory signaling within the network [13, 89, 90]. For efficient online learning in RNNs, other approximations (not necessarily bio-plausible) to RTRL [91–96] have also demonstrated to produce good performance. Given the impressive accuracy achieved by these approximate rules, several studies began to investigate their convergence properties [97], e.g. for random backpropagation weights in feedforward networks [98, 99]. However, the trend of generalization capabilities of these rules, especially in RNNs, is underexamined.

2.2 Loss landscape curvature and generalization performance

Given the central importance of understanding how neural networks perform in situations unseen during training [4, 100–106], the deep learning community has made tremendous efforts to develop tools for understanding generalization that we leverage here. That flat minima could lead to better generalization was observed more than two decades ago [107]. Intuitively, under the same amount of perturbation ϵ in parameter space (e.g. loss landscape changes due to addition of new data) worse performance degradation will be seen around the narrower minima (see Figure 1C). We note that perturbations in parameter space can be linked to that in the input space [6]. Recently, many empirical studies have consistently supported the usefulness of this predictor [108–115]. In particular, the authors of [5] performed an extensive study and found that flatness-based measures have a higher correlation with generalization than other alternatives. Motivated by this, several studies have characterized properties of the loss functions's Hessian — whose eigenspectrum carries information about curvature [116–120]. Connections between flatness and generalization performance have shed light on the reason for greater generalization gaps in large batch training [111, 121–124], and also have inspired optimization methods to favor flatter minima [108, 109, 125–132]. Despite the criticism of scale-dependence of flatness [133], where parameter rescaling can drastically change flatness but not always generalization quality, flatness — with parameter scales taken into account [134, 135] — are connected to PAC-Bayesian generalization error bounds [7–9]. Moreover, a recent theoretical study rigorously connects flatness of the loss surface to generalization in classification tasks under the assumption that labels are (approximately) locally constant [6]. Leveraging the great progress from deep learning community, we aim to study generalization properties of bio-plausible learning rules from a geometric perspective.

3 Results

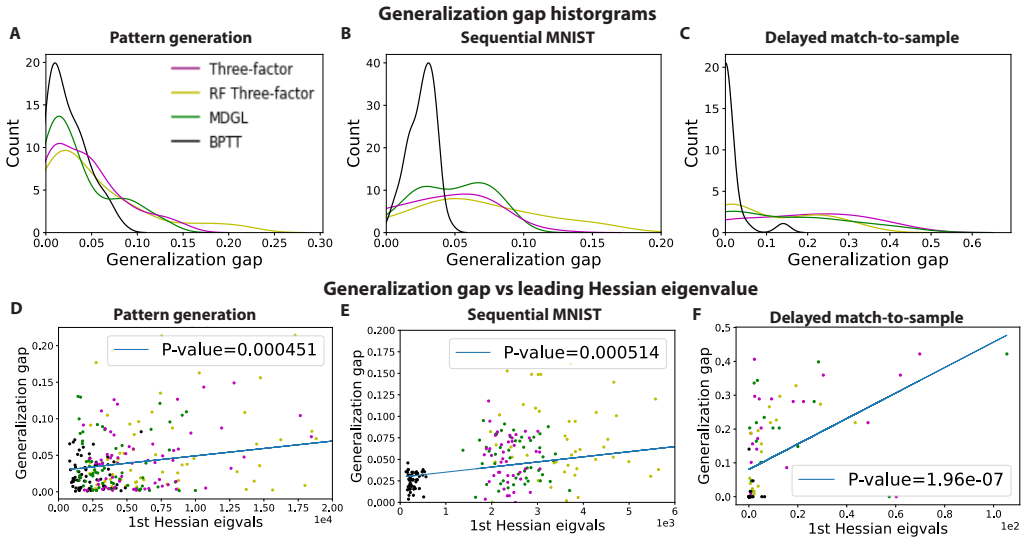

Figure 2: **Bio-plausible temporal credit assignment rules show worse and more variable generalization gap, which can be informed by loss landscape curvature**. A-C) Generalization gap distributions computed at *the end of training* across different random weight initializations for several well-known neuroscience and machine learning tasks. The higher the generalization gap, the worse the generalization performance. BPTT (black), bio-plausible alternatives (magenta, yellow and green). D-F) Scatter plots showing the trend of generalization gap v.s. leading loss Hessian eigenvalue across many runs; each point corresponds to a single run of the same runs as in A-C.

3.1 Network and learning setup

The detailed governing equations of our setup can be found in Methods (Appendix A). We consider a RNN with N_{in} input units, N hidden units and N_{out} readout units (Figure 1A). We verified that trends hold for different network sizes and refer the reader to Appendix A.3 for more details. The update formula for $h_t \in \mathbb{R}^N$ (the hidden state at time t) is governed by:

$$h_{t+1} = \phi(W_h f(h_t), W_x x_t), \tag{1}$$

where $\phi(\cdot) : \mathbb{R}^N \to \mathbb{R}^N$ is the hidden state update function, $f(\cdot) : \mathbb{R}^N \to \mathbb{R}^N$ is the activation function, $W_h \in \mathbb{R}^{N\times N}$ (resp. $W_x \in \mathbb{R}^{N_{in}\times N}$) is the recurrent (resp. input) weight matrix and $x \in \mathbb{R}^{N_{in}}$ is the input. For ϕ, we consider a discrete-time implementation of a rate-based recurrent neural network (RNN) similar to the form in [136] (details in Appendix A). Readout $\hat{y} \in \mathbb{R}^{N_{out}}$, with readout weights $w \in \mathbb{R}^{N_{out}\times N}$, is defined as

$$\hat{y} = \langle w, f(h_t)\rangle. \tag{2}$$

We performed experiments on three tasks: sequential MNIST [137], pattern generation [138] and delayed match-to-sample tasks [139]. The objective is to minimize scalar loss $L \in \mathbb{R}$, which is defined as

$$L(W_h) = \begin{cases} \frac{1}{2TB}\sum_{i=1}^{B}\sum_{t=1}^{T}\sum_{k=1}^{N_{out}}(\hat{y}_{k,t}^{(i)} - y_{k,t}^{(i)})^2, & \text{for regression tasks} \\ \frac{-1}{TB}\sum_{i=1}^{B}\sum_{t=1}^{T}\sum_{k=1}^{N_{out}}\pi_{k,t}^{(i)} log\hat{\pi}_{k,t}^{(i)}, & \text{for classification tasks} \end{cases} \tag{3}$$

given target readout $y \in \mathbb{R}^{N_{out}}$, task duration $T \in \mathbb{R}$ and batch size $B \in \mathbb{R}$. $\pi_{k,t} \in \mathbb{R}$ is the one-hot encoded target for readout unit k at time t and $\hat{\pi}_{k,t} = \text{softmax}_k(\hat{y}_{1,t}, \ldots, \hat{y}_{N_{OUT},t}) = \exp(\hat{y}_{k,t})/\sum_{k'}\exp(\hat{y}_{k',t})$ is the predicted category probability.

Different learning algorithms examined in this work are BPTT (our benchmark), which update weights by computing the exact gradient ($\nabla L(W_h) \in \mathbb{R}^{N\times N}$):

$$\Delta W_h = -\eta\nabla L(W_h), \tag{4}$$

and three SoTA bio-plausible learning rules that update weights using approximate gradient:

$$\widehat{\Delta W_h} = -\eta \tilde{\nabla} L(W_h), \tag{5}$$

where $\tilde{\nabla} L(W_h) \in \mathbb{R}^{N \times N}$ denotes a gradient approximation and $\eta \in \mathbb{R}$ denotes the learning rate. These three learning rules are explained further in Appendix A (Methods) but we note that these bio-plausible learning rules are based on truncations of dependency paths — on the computational graph for the exact gradient— that are biologically implausible to compute (Figure 1A). In all figures, learning rules are labelled as "Three-factor" (symmetric e-prop), "RF Three-factor" (RFLO) and "MDGL", respectively. We remark that the focus here is on comparing artificial to bio-plausible learning rules, rather than between biological rules. Finally, we note that tasks were learned with mostly comparable training accuracies for all learning rules, and that generalization gaps reflect testing departure from these values. We refer the reader to Appendix for more details (Appendix A.3).

3.2 Generalization gap and loss landscape curvature

To study generalization performance, our first step is to compute the generalization gap empirically. Generalization gap is defined train accuracy minus test accuracy; the larger the generalization gap, the worse the generalization performance. For various learning rules, we plot the histogram of generalization gap at the end of training across runs with distinct initializations; different colors represent different learning rules (Figure 2A-C). Notice that these bio-plausible rules achieve worse and more variable generalization performance than their machine learning counterpart (BPTT).

We now investigate if the generalization gap behavior described above correlates well with Loss landscape curvature. We use the leading eigenvalue of the loss' Hessian (where derivatives are taken with respect to model parameters) as a measure for curvature following previous practice [121] and note that it is practical for both empirical [118] and theoretical analyses (Theorem 1). There exists other measures for flatness and due to the scale-dependence issue of Hessian spectrum [133], we also test using parameter scale-independent measures (see Appendix Figures 7 and 8). When we plot generalization gap points from Figure 2A-C against the corresponding leading Hessian eigenvalue, a statistically significant correlation is observed (Figure 2D-F). We also observed such correlation correlation for runs with the learning rule fixed (Appendix Figure 12). We note that we did not expect this relation to be very tight since in addition to worse generalization gap on average, bio-plausible learning rules exhibit increased variance. This is an important and consistent trend we observed but might have been overlooked in previous studies.

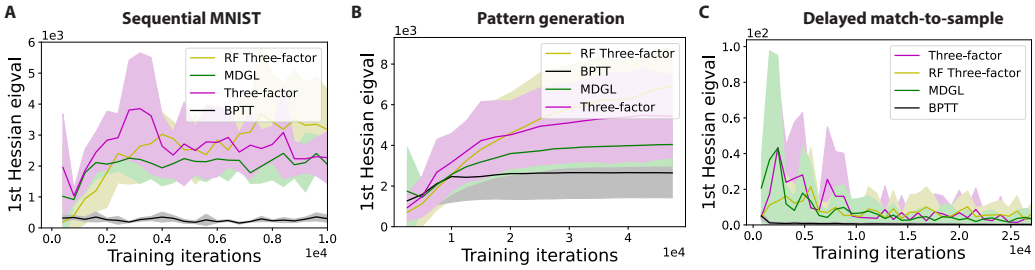

Figure 3: **Bio-plausible gradient approximations tend to approach high curvature regions in loss landscape**. Dominant Hessian eigenvalues are plotted throughout training for bio-plausible learning rules and BPTT. This analysis is done for A) sequential MNIST, B) pattern generation and C) delayed match-to-sample tasks. Solid lines/shaded regions: mean/standard deviation of dominant Hessian eigenvalue curves across five independent runs.

So far, we investigated the endpoints of optimization trajectories, where training performance has converged. Now, we visualize the whole training trajectory. This done is for two reasons: 1) account for early stopping that can halt training anywhere along the trajectory due to time constraints; 2) flatness during training could give indications of avoiding or escaping high-curvature regions. We observe that the biologically motivated gradient approximations tend to rapidly approach high-curvature regions compared to their machine learning counterpart (Figure 3). Together, these results demonstrate a clear trend and a link between generalization gap and loss landscape curvature: both being increased and more variable for bio-plausible rules. We also stopped BPTT early to match the

test accuracy of three-factor rule in Appendix Table 1, and observed similar trends. We also note that the curvature convergence behavior seems to be a shared problem of temporal truncations of the gradient (Appendix Figure 9), which is what the existing bio-plausible gradient approximations for RNNs are based on. Next, we provide a theoretical argument as to why truncated temporal credit assignment rules favor high curvature regions of the loss landscape.

3.3 Theoretical analysis: link between curvature and gradient approximation error

We discussed how generalization can be linked to curvature, and we now examine the link between curvature and gradient alignment during learning dynamics. We represent approximate gradients in a *rule agnostic* manner, where an arbitrary approximation is represented in terms of its component along the gradient direction plus an arbitrary orthogonal vector (Figure 4A):

$$\vec{\hat{g}} = \rho\vec{g} + \vec{e}, \tag{6}$$

where $\vec{g} := \nabla L(W_h) \in \mathbb{R}^{N^2}$ and $\vec{\hat{g}} \in \mathbb{R}^{N^2}$ are the exact and approximate gradients, respectively (we reshaped $\nabla L(W_h)$ into a vector here); $\vec{e} \in \mathbb{R}^{N^2}$ is an arbitrary orthogonal vector to $\vec{g}$. Here, scalar $\rho \in \mathbb{R}$ represents the relative step length along the gradient direction that the approximate rule is making. As we will see in Theorem 1, ρ is an important quantity in our analysis. One can easily compute ρ from $\vec{g}$ and $\vec{\hat{g}}$ by $\rho = \frac{\vec{\hat{g}}^T\vec{g}}{\vec{g}^T\vec{g}}$.

We express weight updates as discrete dynamical systems (with weights W_h as the state variables):

$$W_h^+ \leftarrow W_h^- + F(W_h^-) = W_h^- - \eta\nabla L(W_h^-), \text{ for BPTT} \tag{7}$$

$$W_h^+ \leftarrow W_h^- + \hat{F}(W_h^-) = W_h^- - \eta\tilde{\nabla} L(W_h^-), \text{ for an approximate rule,} \tag{8}$$

where η is the learning rate, $\tilde{\nabla}$ is an approximate gradient, and $F : \mathbb{R}^{N^2} \to \mathbb{R}^{N^2}$ (resp. $\hat{F} : \mathbb{R}^{N^2} \to \mathbb{R}^{N^2}$) denotes the map defined by BPTT (resp. an approximate) weight update rule. Notation W^+ and W^- denotes W at the next and current step, respectively. We note in passing that dynamical systems view of weight updates have been used previously [140, 141].

We introduce additional notations before presenting Theorem 1. $J \in \mathbb{R}^{N^2\times N^2}$ (resp. $\widehat{J} \in \mathbb{R}^{N^2\times N^2}$) is the Jacobian of the dynamical system of BPTT (resp. an approximate rule) in Eq. 7 (resp. Eq. 8). $\lambda_1^J \in \mathbb{R}$ (resp. $\widehat{\lambda_1^J} \in \mathbb{R}$) is the leading eigenvalue for BPTT (resp. approximate rule) Jacobian. $\lambda_1^H \in \mathbb{R}$ is the leading eigenvalue of the loss' Hessian matrix. $W_B^* \in \mathbb{R}^{N\times N}$ (resp. $W_e^* \in \mathbb{R}^{N\times N}$) is the final fixed point for BPTT (resp. an approximate rule). We now present Theorem 1.

Theorem 1. *Consider a RNN defined in Eq. 1 with a single scalar output $\hat{y}$ and least squares loss as in Eq. 3 presented only at the last time step T, and weights are updated according to the difference equation for BPTT 7 (resp. approximate rule 8) on a single example (stochastic gradient descent) using learning rate η_B (resp. η_e). In the limit of stable fixed point convergence with zero training error, the dominant loss' Hessian eigenvalue attained by BPTT (resp. approximate rule) is bounded by $|\lambda_1^H(W_B^*)| < \frac{1}{\eta_B}$ (resp. $|\lambda_1^H(W_e^*)| < \frac{1}{|\rho|\eta_e}$).*

Proof. Full proof is in Appendix B. Here are the main steps involved in the proof:

1. Observe that the *Jacobian* of BPTT discrete time dynamical system (Eq. 7) is precisely the *loss' Hessian* scaled by a constant
2. Using the above relationship, we can bound $|\lambda_1^H|$ from $|\lambda_1^J| < 1$, which is the condition for discrete time dynamical system to converge to a fixed point.
3. However, the link between the Jacobian of an approximate update rule and the loss' Hessian is less obvious. Thus, we derive a link between $|\widehat{\lambda_1^J}|$ and $|\lambda_1^J|$, and then apply the step above.

□

The consequence of the upper bound derived in Theorem 1 is that truncated gradient rules can converge to minima with a higher dominant Hessian eigenvalue than BPTT, with the leading eigenvalue bound

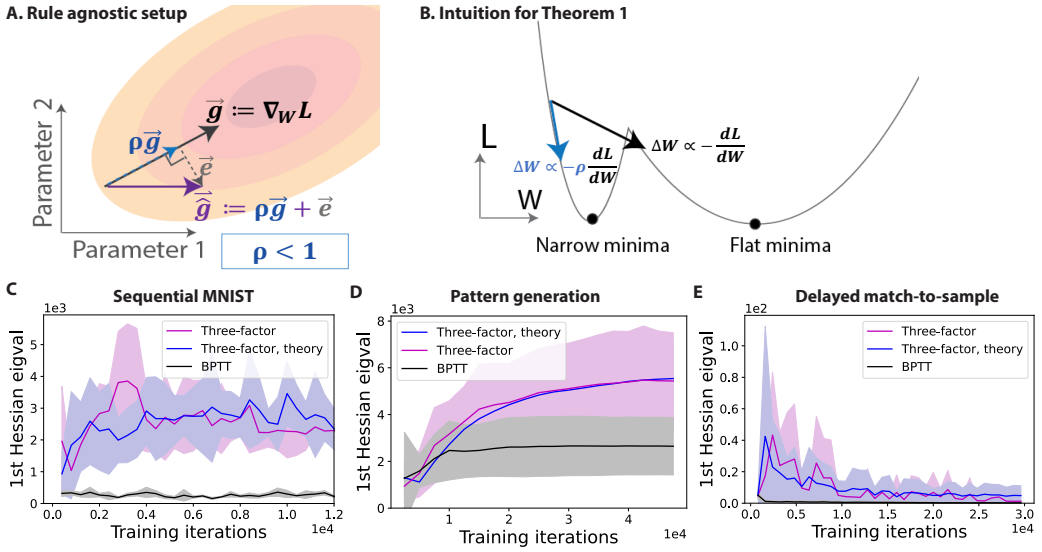

Figure 4: **Preference for high curvature regions connected to worse gradient alignment (rule agnostic analysis)**. A) An arbitrary gradient approximation, $\hat{\vec{g}}$, its component along the gradient direction, $\rho\vec{g}$, and orthogonal to the gradient $\vec{e}$ (Eq. 6). The scalar ρ represents the relative step length along the gradient direction. B) Illustration for Theorem 1: if gradient $\vec{g}$ is aligned with the sharpest directions but error $\vec{e}$ is not (see Appendix Figure 10), smaller step length along the gradient direction can make it harder to step over narrow minima. C-E) Leading loss Hessian eigenvalue v.s. training iteration for three-factor rule, BPTT, and modified BPTT (three-factor, theory). For the latter, step length along the gradient direction of BPTT was matched to three-factor rule by multiplying BPTT update with a factor of ρ and recovers curvature trends of three-factor rule.

inversely proportional to $|\rho|$ ($|\rho| < 1$ usually). In practice, ρ can vary depending on task settings and in our setup, we observed it to be somewhere between 0.02 and 0.3. We remark that this higher upper-bound is consistent with the increased spread of curvature and generalization observed for bio-plausible rules in experiments. Theorem 1 highlights scalar ρ (Eq. 6), relative step length along the gradient direction, as an important factor in the curvature bound. To test that, we eliminate the factor of ρ by reducing learning rule of BPTT such that its step length is matched to that of the three-factor rule. This resulted in the blue curves in Figure 4, which is still trained using BPTT but with the update scaled by a factor of ρ. By matching the step length of BPTT and a three-factor rule along the gradient direction, similar convergence behaviors were observed. Similar observations were also made when the matching step experiment were repeated at three times the learning rate (Appendix Figure 12C) **This result then attributes the curvature preference behavior to relative step length along the gradient direction, and thereby indicating a link between curvature and gradient alignment**. Consistent with earlier results, when the step length of BPTT is matched to that of a three-factor rule, its generalization performance also worsened (Appendix Figure 6).

We make a few more remarks regarding Theorem 1. ρ is usually less than 1 because otherwise the additional orthogonal component $\vec{e}$ would imply a larger update size for the approximate rule compared to BPTT. This would make the approximate rule more prone to numerical instabilities (Eq. 9) and would require a change in solver hyperparameters such as overall learning rate which in turn, influence the update size. In our experiments, when three-factor rule is scaled up to match the along-gradient update sizes of BPTT, we quickly ran into numerical overflow (values of NaNs in the network), which is expected when a very large learning rate is used. The exact point for when this numerical instability is reached depends on many factors such as the model, the task, numerical precision as well as the consistency of alignment between weight and update vectors. We've also equated numerical instabilities in simulations as a proxy for situations problematic for the brain in Discussion. Curiously, $\vec{e}$ did not seem to play a helpful role in finding flatter minima; we provide the reasoning for that in Figure 4B and Appendix Figure 10. Despite making assumptions including scalar output, MSE loss and loss available at the last step, our empirical results indicate that our

conclusions also extend to other setups: vector output, cross-entropy loss and loss that accumulates over time steps (Figure 4). We discuss the generality of Theorem 1 in Appendix B.

One might assume increasing learning rate of the approximate rules could compensate for the reduced step length due to ρ. However, this can raise other issues. Suppose $\Delta W = -\eta\vec{g}$ for exact gradient and we increase the learning rate of approximate update until $\widehat{\Delta W} = -\eta\vec{g} - \eta\vec{e}$ (magnitude of $\vec{e}$ is scaled accordingly), and because $\vec{g} \perp \vec{e}$:

$$\|\widehat{\Delta W}\|^2 = \eta^2\|\vec{g}\|^2 + \eta^2\|\vec{e}\|^2 > \eta^2\|g\|^2 = \|\Delta W\|^2 \tag{9}$$

In other words, if approximate updates $\widehat{\Delta W}$ make the same amount of progress as BPTT along the gradient direction, $\widehat{\Delta W}$ would have a larger magnitude due to the orthogonal component $\vec{e}$, and large update magnitude can be very problematic for numerical stability [142]. Thus, the magnitude of $\vec{e}$ limits the learning rate that can be used. Because of the numerical issues associated with increasing the learning rate for such rules, the differences in generalization and curvature convergence between learning rules cannot be reduced by fine-tuning the learning rate. To balance between this stability issue and the potential benefit of large learning rates, one can consider using a large learning rate early in training to prevent premature stabilization in sharp minima followed by gradual decay to mitigate the stability issue (see Appendix Figure 5).

Taken together, Theorem 1 and Eq. 9 connect gradient alignment with the curvature of converged solution. For an approximation that is aligned poorly with the exact gradient, large orthogonal approximation error vector $\vec{e}$ limit the step length for numerical stability reasons (Eq. 9) and small relative step length ρ correspond to a larger curvature bound (Theorem 1).

4 Conclusion

While developing bio-plausible learning rules is of an interest for both answering neuroscience questions and searching for more efficient training strategies for artificial networks, generalization properties of solutions found by these rules are severely underexamined. Through various well-known machine learning and working memory tasks, we first demonstrate empirically that existing bio-plausible *temporal credit assignment* rules attain worse generalization performance, which is consistent with their tendency to converge to high-curvature regions in loss landscape. Second, our theoretical analysis offers an explanation for this preference of high curvature regions based on worse alignment to true gradient. This regime corresponds to the situation where the step length along the gradient direction is small and the approximation error vector is large. Finally, we test this theory empirically by matching the relative step length along the gradient direction resulting in similar convergence behavior (Figure 4).

5 Discussion

Our study — a stepping stone toward understanding biological generalization using deep learning methods — raises many exciting questions for future investigations, both on the front of stronger deep learning theory and more sophisticated biological ingredients.

Deep learning theory and its implications: In this study, we investigate generalization properties using loss landscape flatness, a promising predictor with recent rigorous connection to generalization gap [6]. Yet, to what extent can flatness explain generalization is still an open question in deep learning. For instance, curvature is a local measure, which means that its informativeness of robustness against global perturbation is limited. Moreover, the theoretical association between flatness and generalization is provided in the form of upper bound [6, 9] and the bound may not always be tight, which is consistent with more variability in generalization gap for bio-plausible learning rules but offers less predictive power. Despite observing a significant correlation between generalization gap and a curvature-based measure in Figure 2, the relationship appears to be messy, suggesting other factors involved for explaining generalization. Given that developing better predictors of generalization is still a work in progress [4], we anticipate stronger theoretical tools to be applied for studying biological generalization in the future. Our results are also consistent with existing findings linking learning rate to loss landscape curvature in deep networks [123]. In the case of bio-plausible gradient approximations, small step length along the gradient direction cannot be compensated by

increasing the learning rate, as that would inadvertently increase the error vector, causing numerical issues (Eq. 9). Curiously, the approximation error vector $\vec{e}$ did not seem to play a role other than restricting the learning rate. While it is well-known that stochastic gradient noise (SGN) can help with finding flat minima due to the alignment of SGN covariance and Hessian near minima [122, 143–145], that may not apply to approximation error vector $\vec{e}$ resulting primarily from temporal truncation of the gradient (see Figure 1A and Appendix Figure 10). This indicates that noise with different properties (e.g. direction) could affect generalization differently, thereby motivating future investigations into applying our curvature-based analysis to broader range of bio-inspired learning systems and examine biological noise with different properties.

Moreover, our results are closely related to a series of study that touched upon the "catapulting" behavior in learning [146–150], at which the 2nd order Taylor term of loss expansion can dominate over the 1st term, which causes learning to "catapult" into a flatter region that can accommodate the step size. If truncation noise is not aligned with the leading Hessian eigenvectors but aligned with the eigendirections with negligible eigenvalues, then it can only have limited contributions to the 2nd Taylor term. In addition, the orthogonal noise term requires a smaller learning rate to be used to avoid numerical issues in simulations, as explained above. This leads to a weaker 2nd Taylor term relative to 1st for bio-plausible temporal credit assignment rules, which would then increase the threshold for step size stability. This is all consistent with the results on convergence to high-curvature loss landscape regions for bio-plausible temporal credit assignment rules seen in this study.

Toward more detailed biological mechanisms: On the front of more sophisticated biological ingredients that may improve generalization performance, we see two lines of approaches: **1-** develop bio-plausible learning rules that align better with the gradient, as suggested by our rule agnostic analysis (Theorem 1, Figure 4); and **2-** instead of studying learning rules in isolation, consider also other neural circuit components [151–158] that could interact with the learning rule. An important component would be the architecture, including connectivity structure and neuron model, found through evolution [156] (see also [159]). To address our main question, this study varies learning rules while holding data, objective function and architecture constant (see [10]). However, these different components can interact, and more sophisticated architecture can facilitate task learning [159–166]. Given the exploding parameter space resulting from such interactions, we believe it requires careful future analysis and is outside of the scope for this one paper.

Additionally, learning rate modulation [167, 168] could be one of many possible remedies employed by the brain. We conjecture that neuromodulatory mechanisms could be coupled with these learning rules to improve the convergence behavior through our scheduled learning rate experiments (Appendix Figure 5), where an initial high learning rate could prevent the learning trajectory from settling in sharp minima prematurely followed by gradual decay to avoid instabilities. One possible way to realize such learning rate modulation could be through serotonin neurons via uncertainty tracking, where the learning rate is high when the reward prediction error is high (this can happen at the beginning of learning) [169]. Since the authors of [169] showed that inhibiting serotonin led to failure in learning rate modulation, we *conjecture* such inhibition might have an impact on the generalization performance of learning outcomes. On the topic of balancing numerical instabilities and potential advantages of large learning rates, while the analog nature of biology may seem to avoid finite precision representation in digital computers that give rise to numerical instabilities, the same problems that lead to numerical instabilities in digital computers, such as big ranges between quantities added or multiplied, remains an issue for biology since quantities must be stored in noisy activity patterns of neurotransmitter release. Future investigations could investigate potential homeostatic mechanisms that regulate quantities in the "optimal operating range" to avoid such instabilities, thereby enabling larger "learning rates" to be used so as to find flatter minima. Other ingredients for future investigations could include intrinsic noise with certain structures [153, 170, 171]; such as if any biological noise exhibit certain properties (e.g. direction, bias/variance) that would make them more favourable for generalization. Taken together, we hope to see follow up investigations — riding on the rapid advancements both at the front of deep learning theory and sophisticated biological mechanisms — into how the brain attains solutions that generalize.

References

[1] Timothy P Lillicrap and Adam Santoro. Backpropagation through time and the brain. *Current opinion in neurobiology*, 55:82–89, 2019.

[2] Luke Y Prince, Ellen Boven, Roy Henha Eyono, Arna Ghosh, Joe Pemberton, Franz Scherr, Claudia Clopath, Rui Ponte Costa, Wolfgang Maass, Blake A Richards, et al. Ccn gac workshop: Issues with learning in biological recurrent neural networks. *arXiv preprint arXiv:2105.05382*, 2021.

[3] Owen Marschall, Kyunghyun Cho, and Cristina Savin. A unified framework of online learning algorithms for training recurrent neural networks. *Journal of Machine Learning Research*, 21(135):1–34, 2020.

[4] Yiding Jiang, Pierre Foret, Scott Yak, Daniel M Roy, Hossein Mobahi, Gintare Karolina Dziugaite, Samy Bengio, Suriya Gunasekar, Isabelle Guyon, and Behnam Neyshabur. Neurips 2020 competition: Predicting generalization in deep learning. *arXiv preprint arXiv:2012.07976*, 2020.

[5] Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. *arXiv preprint arXiv:1912.02178*, 2019.

[6] Henning Petzka, Michael Kamp, Linara Adilova, Cristian Sminchisescu, and Mario Boley. Relative flatness and generalization. *Advances in Neural Information Processing Systems*, 34, 2021.

[7] Gintare Karolina Dziugaite and Daniel M Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*, 2017.

[8] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. *Advances in neural information processing systems*, 30, 2017.

[9] Yusuke Tsuzuku, Issei Sato, and Masashi Sugiyama. Normalized flat minima: Exploring scale invariant definition of flat minima for neural networks using pac-bayesian analysis. In *International Conference on Machine Learning*, pages 9636–9647. PMLR, 2020.

[10] Blake A Richards, Timothy P Lillicrap, Philippe Beaudoin, Yoshua Bengio, Rafal Bogacz, Amelia Christensen, Claudia Clopath, Rui Ponte Costa, Archy de Berker, Surya Ganguli, et al. A deep learning framework for neuroscience. *Nature neuroscience*, 22(11):1761–1770, 2019.

[11] Guillaume Bellec, Franz Scherr, Anand Subramoney, Elias Hajek, Darjan Salaj, Robert Legenstein, and Wolfgang Maass. A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature Communications*, 11(1):1–15, dec 2020.

[12] James M Murray. Local online learning in recurrent networks with random feedback. *ELife*, 8:e43299, 2019.

[13] Yuhan Helena Liu, Stephen Smith, Stefan Mihalas, Eric Shea-Brown, and Uygar Sümbül. Cell-type–specific neuromodulation guides synaptic credit assignment in a spiking neural network. *Proceedings of the National Academy of Sciences*, 118(51), 2021.

[14] Colin Bredenberg, Benjamin Lyo, Eero Simoncelli, and Cristina Savin. Impression learning: Online representation learning with synaptic plasticity. *Advances in Neural Information Processing Systems*, 34, 2021.

[15] Yanis Bahroun, Dmitri Chklovskii, and Anirvan Sengupta. A normative and biologically plausible algorithm for independent component analysis. *Advances in Neural Information Processing Systems*, 34, 2021.

[16] Basile Confavreux, Friedemann Zenke, Everton Agnes, Timothy Lillicrap, and Tim Vogels. A meta-learning approach to (re) discover plasticity rules that carve a desired function into a neural network. *Advances in Neural Information Processing Systems*, 33:16398–16408, 2020.

[17] Gabriel Ocker and Michael Buice. Tensor decompositions of higher-order correlations by nonlinear hebbian plasticity. *Advances in Neural Information Processing Systems*, 34, 2021.

[18] Danil Tyulmankov, Ching Fang, Annapurna Vadaparty, and Guangyu Robert Yang. Biological key-value memory networks. *Advances in Neural Information Processing Systems*, 34, 2021.

[19] Alexander Meulemans, Matilde Tristany Farinha, Javier Garcia Ordonez, Pau Vilimelis Aceituno, João Sacramento, and Benjamin F Grewe. Credit assignment in neural networks through deep feedback control. *Advances in Neural Information Processing Systems*, 34, 2021.

[20] Brandon McMahan, Michael Kleinman, and Jonathan Kao. Learning rule influences recurrent network representations but not attractor structure in decision-making tasks. *Advances in Neural Information Processing Systems*, 34, 2021.

[21] Paul Haider, Benjamin Ellenberger, Laura Kriener, Jakob Jordan, Walter Senn, and Mihai A Petrovici. Latent equilibrium: A unified learning theory for arbitrarily fast computation with arbitrarily slow neurons. *Advances in Neural Information Processing Systems*, 34:17839–17851, 2021.

[22] Roman Pogodin, Yash Mehta, Timothy Lillicrap, and Peter E Latham. Towards biologically plausible convolutional networks. *Advances in Neural Information Processing Systems*, 34:13924–13936, 2021.

[23] Johannes Friedrich, Siavash Golkar, Shiva Farashahi, Alexander Genkin, Anirvan Sengupta, and Dmitri Chklovskii. Neural optimal feedback control with local learning rules. *Advances in Neural Information Processing Systems*, 34, 2021.

[24] Colin Bredenberg, Eero Simoncelli, and Cristina Savin. Learning efficient task-dependent representations with synaptic plasticity. *Advances in Neural Information Processing Systems*, 33:15714–15724, 2020.

[25] Bernd Illing, Jean Ventura, Guillaume Bellec, and Wulfram Gerstner. Local plasticity rules can learn deep representations using self-supervised contrastive predictions. *Advances in Neural Information Processing Systems*, 34, 2021.

[26] Elias Najarro and Sebastian Risi. Meta-learning through hebbian plasticity in random networks. *Advances in Neural Information Processing Systems*, 33:20719–20731, 2020.

[27] Pablo Tano, Peter Dayan, and Alexandre Pouget. A local temporal difference code for distributional reinforcement learning. *Advances in Neural Information Processing Systems*, 33:13662–13673, 2020.

[28] Lea Duncker, Laura Driscoll, Krishna V Shenoy, Maneesh Sahani, and David Sussillo. Organizing recurrent network dynamics by task-computation to enable continual learning. *Advances in neural information processing systems*, 33:14387–14397, 2020.

[29] Aran Nayebi, Sanjana Srivastava, Surya Ganguli, and Daniel L Yamins. Identifying learning rules from neural network observables. *Advances in Neural Information Processing Systems*, 33:2639–2650, 2020.

[30] Zoe Ashwood, Nicholas A Roy, Ji Hyun Bak, and Jonathan W Pillow. Inferring learning rules from animal decision-making. *Advances in Neural Information Processing Systems*, 33:3442–3453, 2020.

[31] Siavash Golkar, David Lipshutz, Yanis Bahroun, Anirvan Sengupta, and Dmitri Chklovskii. A simple normative network approximates local non-hebbian learning in the cortex. *Advances in neural information processing systems*, 33:7283–7295, 2020.

[32] Daniel R Kepple, Rainer Engelken, and Kanaka Rajan. Curriculum learning as a tool to uncover learning principles in the brain. In *International Conference on Learning Representations*, 2021.

[33] Danil Tyulmankov, Guangyu Robert Yang, and LF Abbott. Meta-learning synaptic plasticity and memory addressing for continual familiarity detection. *Neuron*, 110(3):544–557, 2022.

[34] Eli Moore and Rishidev Chaudhuri. Using noise to probe recurrent neural network structure and prune synapses. *Advances in Neural Information Processing Systems*, 33:14046–14057, 2020.

[35] Grace Wan Yu Ang, Clara S Tang, Y Audrey Hay, Sara Zannone, Ole Paulsen, and Claudia Clopath. The functional role of sequentially neuromodulated synaptic plasticity in behavioural learning. *PLoS Computational Biology*, 17(6):e1009017, 2021.

[36] Eren Sezener, Agnieszka Grabska-Barwińska, Dimitar Kostadinov, Maxime Beau, Sanjukta Krishnagopal, David Budden, Marcus Hutter, Joel Veness, Matthew Botvinick, Claudia Clopath, et al. A rapid and efficient learning rule for biological neural circuits. *BioRxiv*, 2021.

[37] Owen Marschall, Kyunghyun Cho, and Cristina Savin. Using local plasticity rules to train recurrent neural networks. *arXiv preprint arXiv:1905.12100*, 2019.

[38] David Clark, LF Abbott, and SueYeon Chung. Credit assignment through broadcasting a global error vector. *Advances in Neural Information Processing Systems*, 34:10053–10066, 2021.

[39] Maxence Ernoult, Fabrice Normandin, Abhinav Moudgil, Sean Spinney, Eugene Belilovsky, Irina Rish, Blake Richards, and Yoshua Bengio. Towards scaling difference target propagation by learning backprop targets. *arXiv preprint arXiv:2201.13415*, 2022.

[40] Jack Lindsey and Ashok Litwin-Kumar. Learning to learn with feedback and local plasticity. *Advances in Neural Information Processing Systems*, 33:21213–21223, 2020.

[41] Sukbin Lim, Jillian L McKee, Luke Woloszyn, Yali Amit, David J Freedman, David L Sheinberg, and Nicolas Brunel. Inferring learning rules from distributions of firing rates in cortical neurons. *Nature neuroscience*, 18(12):1804–1810, 2015.

[42] Artur Luczak, Bruce L. McNaughton, and Yoshimasa Kubo. Neurons learn by predicting future activity. *Nature Machine Intelligence*, 4(1):62–72, January 2022.

[43] James C.R. Whittington and Rafal Bogacz. Theories of Error Back-Propagation in the Brain. *Trends in Cognitive Sciences*, 2019.

[44] Saurabh Vyas, Matthew D Golub, David Sussillo, and Krishna V Shenoy. Computation through neural population dynamics. *Annual Review of Neuroscience*, 43:249–275, 2020.

[45] Matthew G Perich, Charlotte Arlt, Sofia Soares, Megan E Young, Clayton P Mosher, Juri Minxha, Eugene Carter, Ueli Rutishauser, Peter H Rudebeck, Christopher D Harvey, et al. Inferring brain-wide interactions using data-constrained recurrent neural network models. *bioRxiv*, pages 2020–12, 2021.

[46] Jimmy Smith, Scott Linderman, and David Sussillo. Reverse engineering recurrent neural networks with jacobian switching linear dynamical systems. *Advances in Neural Information Processing Systems*, 34, 2021.

[47] Michael Kleinman, Chandramouli Chandrasekaran, and Jonathan Kao. A mechanistic multi-area recurrent network model of decision-making. *Advances in Neural Information Processing Systems*, 34, 2021.

[48] Friedrich Schuessler, Francesca Mastrogiuseppe, Alexis Dubreuil, Srdjan Ostojic, and Omri Barak. The interplay between randomness and structure during learning in rnns. *Advances in neural information processing systems*, 33:13352–13362, 2020.

[49] Rylan Schaeffer, Mikail Khona, Leenoy Meshulam, Ila Rani Fiete, et al. Reverse-engineering recurrent neural network solutions to a hierarchical inference task for mice. *bioRxiv*, 2020.

[50] Guangyu Robert Yang, Madhura R Joglekar, H Francis Song, William T Newsome, and Xiao-Jing Wang. Task representations in neural networks trained to perform many cognitive tasks. *Nature neuroscience*, 22(2):297–306, 2019.

[51] Valerio Mante, David Sussillo, Krishna V Shenoy, and William T Newsome. Context-dependent computation by recurrent dynamics in prefrontal cortex. *nature*, 503(7474):78–84, 2013.

[52] Joshua Glaser, Matthew Whiteway, John P Cunningham, Liam Paninski, and Scott Linderman. Recurrent switching dynamical systems models for multiple interacting neural populations. *Advances in neural information processing systems*, 33:14867–14878, 2020.

[53] Jonathan Kadmon, Jonathan Timcheck, and Surya Ganguli. Predictive coding in balanced neural networks with noise, chaos and delays. *Advances in neural information processing systems*, 33:16677–16688, 2020.

[54] Jonathan Dong, Ruben Ohana, Mushegh Rafayelyan, and Florent Krzakala. Reservoir computing meets recurrent kernels and structured transforms. *Advances in Neural Information Processing Systems*, 33:16785–16796, 2020.

[55] Elia Turner, Kabir V Dabholkar, and Omri Barak. Charting and navigating the space of solutions for recurrent neural networks. *Advances in Neural Information Processing Systems*, 34:25320–25333, 2021.

[56] Sandra Nestler, Christian Keup, David Dahmen, Matthieu Gilson, Holger Rauhut, and Moritz Helias. Unfolding recurrence by green's functions for optimized reservoir computing. *Advances in Neural Information Processing Systems*, 33:17380–17390, 2020.

[57] Robert Kim and Terrence J Sejnowski. Strong inhibitory signaling underlies stable temporal dynamics and working memory in spiking neural networks. *Nature Neuroscience*, 24(1):129–139, 2021.

[58] Adrian Valente, Srdjan Ostojic, and Jonathan Pillow. Probing the relationship between linear dynamical systems and low-rank recurrent neural network models. *arXiv preprint arXiv:2110.09804*, 2021.

[59] Tom Macpherson, Anne Churchland, Terry Sejnowski, James DiCarlo, Yukiyasu Kamitani, Hidehiko Takahashi, and Takatoshi Hikida. Natural and artificial intelligence: A brief introduction to the interplay between ai and neuroscience research. *Neural Networks*, 144:603–613, 2021.

[60] Ben Tsuda, Stefan C Pate, Kay M Tye, Hava T Siegelmann, and Terrence J Sejnowski. Neuromodulators generate multiple context-relevant behaviors in a recurrent neural network by shifting activity hypertubes. *bioRxiv*, pages 2021–05, 2022.

[61] Nuttida Rungratsameetaweemana, Robert Kim, John T Serences, and Terrence J Sejnowski. Probabilistic visual processing in humans and recurrent neural networks. *Journal of Vision*, 22(3):24–24, 2022.

[62] Brian DePasquale, Christopher J Cueva, Kanaka Rajan, G Sean Escola, and LF Abbott. full-force: A target-based method for training recurrent networks. *PloS one*, 13(2):e0191527, 2018.

[63] Christopher Langdon and Tatiana A Engel. Latent circuit inference from heterogeneous neural responses during cognitive tasks. *bioRxiv*, 2022.

[64] R. J. Williams and D. Zipser. Gradient-based learning algorithms for recurrent networks and their computational complexity. In Y. Chauvin and D. E. Rumelhart, editors, *Backpropagation: Theory, Architectures and Applications*, chapter 13, pages 433–486. Hillsdale, NJ: Erlbaum, 1995.

[65] Drew Linsley, Alekh Karkada Ashok, Lakshmi Narasimhan Govindarajan, Rex Liu, and Thomas Serre. Stable and expressive recurrent vision models. *arXiv preprint arXiv:2005.11362*, 2020.

[66] Timothy P Lillicrap, Adam Santoro, Luke Marris, Colin J Akerman, and Geoffrey Hinton. Backpropagation and the brain. *Nature Reviews Neuroscience*, pages 1–12, 2020.

[67] Isabella Pozzi, Sander Bohté, and Pieter Roelfsema. A biologically plausible learning rule for deep learning in the brain. *arXiv preprint arXiv:1811.01768*, 2018.

[68] Axel Laborieux, Maxence Ernoult, Benjamin Scellier, Yoshua Bengio, Julie Grollier, and Damien Querlioz. Scaling equilibrium propagation to deep convnets by drastically reducing its gradient estimator bias. *Frontiers in neuroscience*, 15:129, 2021.

[69] Beren Millidge, Alexander Tschantz, Anil K Seth, and Christopher L Buckley. Activation relaxation: A local dynamical approximation to backpropagation in the brain. *arXiv preprint arXiv:2009.05359*, 2020.

[70] João Sacramento, Rui Ponte Costa, Yoshua Bengio, and Walter Senn. Dendritic cortical microcircuits approximate the backpropagation algorithm. *arXiv preprint arXiv:1810.11393*, 2018.

[71] Yali Amit. Deep learning with asymmetric connections and hebbian updates. *Frontiers in computational neuroscience*, 13:18, 2019.

[72] Jonathan E Rubin, Catalina Vich, Matthew Clapp, Kendra Noneman, and Timothy Verstynen. The credit assignment problem in cortico-basal ganglia-thalamic networks: A review, a problem and a possible solution. *European Journal of Neuroscience*, 53(7):2234–2253, 2021.

[73] Alexandre Payeur, Jordan Guerguiev, Friedemann Zenke, Blake A Richards, and Richard Naud. Burst-dependent synaptic plasticity can coordinate learning in hierarchical circuits. *Nature neuroscience*, pages 1–10, 2021.

[74] Pieter R. Roelfsema and Anthony Holtmaat. Control of synaptic plasticity in deep cortical networks. *Nature Reviews Neuroscience*, 19(3):166–180, feb 2018.

[75] Timothy P Lillicrap, Daniel Cownden, Douglas B Tweed, and Colin J Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature communications*, 7(1):1–10, 2016.

[76] Jeffrey C. Magee and Christine Grienberger. Synaptic Plasticity Forms and Functions. *Annual Review of Neuroscience*, 43(1):95–117, jul 2020.

[77] Wulfram Gerstner, Marco Lehmann, Vasiliki Liakoni, Dane Corneil, and Johanni Brea. Eligibility Traces and Plasticity on Behavioral Time Scales: Experimental Support of NeoHebbian Three-Factor Learning Rules. *Frontiers in Neural Circuits*, 12:53, jul 2018.

[78] Magdalena Sanhueza and John Lisman. The CaMKII/NMDAR complex as a molecular memory. *Molecular Brain*, 6(1):10, feb 2013.

[79] Stijn Cassenaer and Gilles Laurent. Conditional modulation of spike-timing-dependent plasticity for olfactory learning. *Nature*, 482(7383):47–51, feb 2012.

[80] Sho Yagishita, Akiko Hayashi-Takagi, Graham C.R. Ellis-Davies, Hidetoshi Urakubo, Shin Ishii, and Haruo Kasai. A critical time window for dopamine actions on the structural plasticity of dendritic spines. *Science*, 345(6204):1616–1620, sep 2014.

[81] Aparna Suvrathan. Beyond STDP — towards diverse and functionally relevant plasticity rules. *Current Opinion in Neurobiology*, 54:12–19, feb 2019.

[82] Wolfram Schultz. Dopamine reward prediction-error signalling: a two-component response. *Nature Reviews Neuroscience*, 17(3):183, 2016.

[83] Johnatan Aljadeff, James D'amour, Rachel E Field, Robert C Froemke, and Claudia Clopath. Cortical credit assignment by hebbian, neuromodulatory and inhibitory plasticity. *arXiv preprint arXiv:1911.00307*, 2019.

[84] Răzvan V Florian. Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity. *Neural computation*, 19(6):1468–1502, 2007.

[85] Michael A. Farries and Adrienne L. Fairhall. Reinforcement Learning With Modulated Spike Timing–Dependent Synaptic Plasticity. *Journal of Neurophysiology*, 98(6):3648–3665, dec 2007.

[86] Robert Legenstein, Dejan Pecevski, and Wolfgang Maass. A learning theory for reward-modulated spike-timing-dependent plasticity with application to biofeedback. *PLoS computational biology*, 4(10):e1000180, 2008.

[87] Zuzanna Brzosko, Susanna B Mierau, and Ole Paulsen. Neuromodulation of spike-timing-dependent plasticity: past, present, and future. *Neuron*, 103(4):563–581, 2019.

[88] Verena Pawlak, Jeffery R Wickens, Alfredo Kirkwood, and Jason ND Kerr. Timing is not everything: neuromodulation opens the stdp gate. *Frontiers in synaptic neuroscience*, 2:146, 2010.

[89] Stephen J Smith, Uygar Sümbül, Lucas T Graybuck, Forrest Collman, Sharmishtaa Seshamani, Rohan Gala, Olga Gliko, Leila Elabbady, Jeremy A Miller, Trygve E Bakken, et al. Single-cell transcriptomic evidence for dense intracortical neuropeptide networks. *Elife*, 8:e47889, 2019.

[90] Stephen J Smith, Michael Hawrylycz, Jean Rossier, and Uygar Sümbül. New light on cortical neuropeptides and synaptic network plasticity. *Current Opinion in Neurobiology*, 63:176–188, 2020.

[91] Asier Mujika, Florian Meier, and Angelika Steger. Approximating real-time recurrent learning with random kronecker factors. *Advances in Neural Information Processing Systems*, 31, 2018.

[92] Corentin Tallec and Yann Ollivier. Unbiased online recurrent optimization. *arXiv preprint arXiv:1702.05043*, 2017.

[93] Tim Cooijmans and James Martens. On the variance of unbiased online recurrent optimization. *arXiv preprint arXiv:1902.02405*, 2019.

[94] Christopher Roth, Ingmar Kanitscheider, and Ila Fiete. Kernel rnn learning (kernl). In *International Conference on Learning Representations*, 2018.

[95] Jacob Menick, Erich Elsen, Utku Evci, Simon Osindero, Karen Simonyan, and Alex Graves. A practical sparse approximation for real time recurrent learning. *arXiv preprint arXiv:2006.07232*, 2020.

[96] Friedemann Zenke and Emre O Neftci. Brain-inspired learning on neuromorphic substrates. *arXiv preprint arXiv:2010.11931*, 2020.

[97] Yinan Cao, Christopher Summerfield, and Andrew Saxe. Characterizing emergent representations in a space of candidate learning rules for deep networks. *Advances in Neural Information Processing Systems*, 33:8660–8670, 2020.

[98] Ganlin Song, Ruitu Xu, and John Lafferty. Convergence and alignment of gradient descent with random backpropagation weights. *Advances in Neural Information Processing Systems*, 34, 2021.

[99] Manuela Girotti, Ioannis Mitliagkas, and Gauthier Gidel. Convergence analysis and implicit regularization of feedback alignment for deep linear networks. *arXiv preprint arXiv:2110.10815*, 2021.

[100] Huan Xu and Shie Mannor. Robustness and generalization. *Machine learning*, 86(3):391–423, 2012.

[101] Zeyuan Allen-Zhu and Yuanzhi Li. Can sgd learn recurrent neural networks with provable generalization? *Advances in Neural Information Processing Systems*, 32, 2019.

[102] Madhu S Advani, Andrew M Saxe, and Haim Sompolinsky. High-dimensional dynamics of generalization error in neural networks. *Neural Networks*, 132:428–446, 2020.

[103] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

[104] Valentin Thomas, Fabian Pedregosa, Bart Merriënboer, Pierre-Antoine Manzagol, Yoshua Bengio, and Nicolas Le Roux. On the interplay between noise and curvature and its effect on optimization and generalization. In *International Conference on Artificial Intelligence and Statistics*, pages 3503–3513. PMLR, 2020.

[105] Mohammad Pezeshki, Oumar Kaba, Yoshua Bengio, Aaron C Courville, Doina Precup, and Guillaume Lajoie. Gradient starvation: A learning proclivity in neural networks. *Advances in Neural Information Processing Systems*, 34, 2021.

[106] Aristide Baratin, Thomas George, César Laurent, R Devon Hjelm, Guillaume Lajoie, Pascal Vincent, and Simon Lacoste-Julien. Implicit regularization via neural feature alignment. In *International Conference on Artificial Intelligence and Statistics*, pages 2269–2277. PMLR, 2021.

[107] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural computation*, 9(1):1–42, 1997.

[108] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

[109] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124018, 2019.

[110] Xu Sun, Zhiyuan Zhang, Xuancheng Ren, Ruixuan Luo, and Liangyou Li. Exploring the vulnerability of deep neural networks: A study of parameter corruption. *arXiv preprint arXiv:2006.05620*, 2020.

[111] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

[112] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31, 2018.

[113] Yu Feng and Yuhai Tu. The inverse variance–flatness relation in stochastic gradient descent is critical for finding flat minima. *Proceedings of the National Academy of Sciences*, 118(9), 2021.

[114] Roman Novak, Yasaman Bahri, Daniel A Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. Sensitivity and generalization in neural networks: an empirical study. In *International Conference on Learning Representations*, 2018.

[115] Huan Wang, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. Identifying generalization properties in neural networks. *arXiv preprint arXiv:1809.07402*, 2018.

[116] Levent Sagun, Utku Evci, V Ugur Guney, Yann Dauphin, and Leon Bottou. Empirical analysis of the hessian of over-parametrized neural networks. *arXiv preprint arXiv:1706.04454*, 2017.

[117] Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via hessian eigenvalue density. In *International Conference on Machine Learning*, pages 2232–2241. PMLR, 2019.

[118] Zhewei Yao, Amir Gholami, Kurt Keutzer, and Michael W Mahoney. Pyhessian: Neural networks through the lens of the hessian. In *2020 IEEE international conference on big data (Big data)*, pages 581–590. IEEE, 2020.

[119] Zhenyu Liao and Michael W Mahoney. Hessian eigenspectra of more realistic nonlinear models. *Advances in Neural Information Processing Systems*, 34, 2021.

[120] Zeke Xie, Qian-Yuan Tang, Yunfeng Cai, Mingming Sun, and Ping Li. On the power-law spectrum in deep learning: A bridge to protein science. *arXiv preprint arXiv:2201.13011*, 2022.

[121] Zhewei Yao, Amir Gholami, Qi Lei, Kurt Keutzer, and Michael W Mahoney. Hessian-based analysis of large batch training and robustness to adversaries. *Advances in Neural Information Processing Systems*, 31, 2018.

[122] Zeke Xie, Issei Sato, and Masashi Sugiyama. A diffusion theory for deep learning dynamics: Stochastic gradient descent exponentially favors flat minima. *arXiv preprint arXiv:2002.03495*, 2020.

[123] Stanisław Jastrzębski, Zachary Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. Finding flatter minima with sgd. 2018.

[124] Chiyuan Zhang, Qianli Liao, Alexander Rakhlin, Brando Miranda, Noah Golowich, and Tomaso Poggio. Theory of deep learning iib: Optimization properties of sgd. *arXiv preprint arXiv:1801.02254*, 2018.

[125] Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. *Advances in neural information processing systems*, 30, 2017.

[126] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

[127] Carlo Baldassi, Fabrizio Pittorino, and Riccardo Zecchina. Shaping the learning landscape in neural networks around wide flat minima. *Proceedings of the National Academy of Sciences*, 117(1):161–170, 2020.

[128] Jean Kaddour, Linqing Liu, Ricardo Silva, and Matt J Kusner. Questions for flat-minima optimization of modern neural networks. *arXiv preprint arXiv:2202.00661*, 2022.

[129] Yaowei Zheng, Richong Zhang, and Yongyi Mao. Regularizing neural networks via adversarial model perturbation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8156–8165, 2021.

[130] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. *Advances in Neural Information Processing Systems*, 33:2958–2969, 2020.

[131] Ryo Karakida, Shotaro Akaho, and Shun-ichi Amari. The normalization method for alleviating pathological sharpness in wide neural networks. *Advances in neural information processing systems*, 32, 2019.

[132] Adepu Ravi Sankar, Yash Khasbage, Rahul Vigneswaran, and Vineeth N Balasubramanian. A deeper look at the hessian eigenspectrum of deep neural networks and its applications to regularization. *arXiv preprint arXiv:2012.03801*, 2020.

[133] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *International Conference on Machine Learning*, pages 1019–1028. PMLR, 2017.

[134] Tengyuan Liang, Tomaso Poggio, Alexander Rakhlin, and James Stokes. Fisher-rao metric, geometry, and complexity of neural networks. In *The 22nd international conference on artificial intelligence and statistics*, pages 888–896. PMLR, 2019.

[135] Akshay Rangamani, Nam H Nguyen, Abhishek Kumar, Dzung Phan, Sang H Chin, and Trac D Tran. A scale invariant flatness measure for deep network minima. *arXiv preprint arXiv:1902.02434*, 2019.

[136] Daniel B Ehrlich, Jasmine T Stone, David Brandfonbrener, Alexander Atanasov, and John D Murray. Psychrnn: An accessible and flexible python package for training recurrent neural network models on cognitive tasks. *Eneuro*, 8(1), 2021.

[137] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[138] Wilten Nicola and Claudia Clopath. Supervised learning in spiking neural networks with force training. *Nature communications*, 8(1):1–15, 2017.

[139] Travis Meyer, Xue-Lian Qi, Terrence R Stanford, and Christos Constantinidis. Stimulus selectivity in dorsal and ventral prefrontal cortex after training in working memory tasks. *Journal of neuroscience*, 31(17):6266–6276, 2011.

[140] Andrew M Saxe, James L McClelland, and Surya Ganguli. A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, 116(23):11537–11546, 2019.

[141] Gauthier Gidel, Francis Bach, and Simon Lacoste-Julien. Implicit regularization of discrete gradient dynamics in linear neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[142] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[143] Zhanxing Zhu, Jingfeng Wu, Bing Yu, Lei Wu, and Jinwen Ma. The anisotropic noise in stochastic gradient descent: Its behavior of escaping from sharp minima and regularization effects. *arXiv preprint arXiv:1803.00195*, 2018.

[144] Mingwei Wei and David J Schwab. How noise affects the hessian spectrum in overparameterized neural networks. *arXiv preprint arXiv:1910.00195*, 2019.

[145] Samuel L Smith and Quoc V Le. A bayesian perspective on generalization and stochastic gradient descent. *arXiv preprint arXiv:1710.06451*, 2017.

[146] Jeremy M Cohen, Simran Kaur, Yuanzhi Li, J Zico Kolter, and Ameet Talwalkar. Gradient descent on neural networks typically occurs at the edge of stability. *arXiv preprint arXiv:2103.00065*, 2021.

[147] Aitor Lewkowycz, Yasaman Bahri, Ethan Dyer, Jascha Sohl-Dickstein, and Guy Gur-Ari. The large learning rate phase of deep learning: the catapult mechanism. *arXiv preprint arXiv:2003.02218*, 2020.

[148] Stanislaw Jastrzebski, Maciej Szymczak, Stanislav Fort, Devansh Arpit, Jacek Tabor, Kyunghyun Cho, and Krzysztof Geras. The break-even point on optimization trajectories of deep neural networks. *arXiv preprint arXiv:2002.09572*, 2020.

[149] Justin Gilmer, Behrooz Ghorbani, Ankush Garg, Sneha Kudugunta, Behnam Neyshabur, David Cardoze, George Edward Dahl, Zachary Nado, and Orhan Firat. A loss curvature perspective on training instabilities of deep learning models. In *International Conference on Learning Representations*, 2021.

[150] Sanjeev Arora, Zhiyuan Li, and Abhishek Panigrahi. Understanding gradient descent on edge of stability in deep learning. *arXiv preprint arXiv:2205.09745*, 2022.

[151] Vladimir Ivanov and Konstantinos Michmizos. Increasing liquid state machine performance with edge-of-chaos dynamics organized by astrocyte-modulated plasticity. *Advances in Neural Information Processing Systems*, 34, 2021.

[152] Lukas Braun and Tim Vogels. Online learning of neural computations from sparse temporal feedback. *Advances in Neural Information Processing Systems*, 34, 2021.

[153] Joel Dapello, Jenelle Feather, Hang Le, Tiago Marques, David Cox, Josh McDermott, James J DiCarlo, and SueYeon Chung. Neural population geometry reveals the role of stochasticity in robust perception. *Advances in Neural Information Processing Systems*, 34, 2021.

[154] Maurizio De Pittà, Nicolas Brunel, and Andrea Volterra. Astrocytes: Orchestrating synaptic plasticity? *Neuroscience*, 323:43–61, 2016.

[155] James M Murray and G Sean Escola. Remembrance of things practiced with fast and slow learning in cortical and subcortical pathways. *Nature Communications*, 11(1):1–12, 2020.

[156] Anthony M Zador. A critique of pure learning and what artificial neural networks can learn from animal brains. *Nature communications*, 10(1):1–7, 2019.

[157] Dhireesha Kudithipudi, Mario Aguilar-Simon, Jonathan Babb, Maxim Bazhenov, Douglas Blackiston, Josh Bongard, Andrew P Brna, Suraj Chakravarthi Raja, Nick Cheney, Jeff Clune, et al. Biological underpinnings for lifelong learning machines. *Nature Machine Intelligence*, 4(3):196–210, 2022.

[158] Yazan N Billeh, Binghuang Cai, Sergey L Gratiy, Kael Dai, Ramakrishnan Iyer, Nathan W Gouwens, Reza Abbasi-Asl, Xiaoxuan Jia, Joshua H Siegle, Shawn R Olsen, et al. Systematic integration of structural and functional data into multi-scale models of mouse primary visual cortex. *Neuron*, 106(3):388–403, 2020.

[159] Victor Geadah, Stefan Horoi, Giancarlo Kerg, Guy Wolf, and Guillaume Lajoie. Goal-driven optimization of single-neuron properties in artificial networks reveals regularization role of neural diversity and adaptation. *bioRxiv*, 2022.

[160] Ilenna Simone Jones and Konrad Paul Kording. Can single neurons solve mnist? the computational power of biological dendritic trees. *arXiv preprint arXiv:2009.01269*, 2020.

[161] Nicolas Perez-Nieves, Vincent CH Leung, Pier Luigi Dragotti, and Dan FM Goodman. Neural heterogeneity promotes robust learning. *Nature communications*, 12(1):1–9, 2021.

[162] Laureline Logiaco, LF Abbott, and Sean Escola. Thalamic control of cortical dynamics in a model of flexible motor sequencing. *Cell reports*, 35(9):109090, 2021.

[163] Guangyu Robert Yang and Manuel Molano Mazon. Next-generation of recurrent neural network models for cognition. 2021.

[164] Corinne Teeter, Ramakrishnan Iyer, Vilas Menon, Nathan Gouwens, David Feng, Jim Berg, Aaron Szafer, Nicholas Cain, Hongkui Zeng, Michael Hawrylycz, et al. Generalized leaky integrate-and-fire models classify multiple neuron types. *Nature communications*, 9(1):1–15, 2018.

[165] Chloe Winston, Dana Mastrovito, Eric Shea-Brown, and Stefan Mihalas. Heterogeneity in neuronal dynamics is learned by gradient descent for temporal processing tasks. *bioRxiv*, 2022.

[166] Christoph Stöckl, Dominik Lang, and Wolfgang Maass. Probabilistic skeletons endow brain-like neural networks with innate computing capabilities. *bioRxiv*, 2021.

[167] Yuanzhi Li, Colin Wei, and Tengyu Ma. Towards explaining the regularization effect of initial large learning rate in training neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[168] Niru Maheswaranathan, David Sussillo, Luke Metz, Ruoxi Sun, and Jascha Sohl-Dickstein. Reverse engineering learned optimizers reveals known and novel mechanisms. *Advances in Neural Information Processing Systems*, 34, 2021.

[169] Cooper D Grossman, Bilal A Bari, and Jeremiah Y Cohen. Serotonin neurons modulate learning rate through uncertainty. *Current Biology*, 2021.

[170] Antonio Orvieto, Hans Kersting, Frank Proske, Francis Bach, and Aurelien Lucchi. Anti-correlated noise injection for improved generalization. *arXiv preprint arXiv:2202.02831*, 2022.

[171] Mo Zhou, Tianyi Liu, Yan Li, Dachao Lin, Enlu Zhou, and Tuo Zhao. Toward understanding the importance of noise in training neural networks. In *International Conference on Machine Learning*, pages 7594–7602. PMLR, 2019.

[172] Stanisław Jastrzębski, Zachary Kenton, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. On the relation between the sharpest directions of dnn loss and the sgd step length. *arXiv preprint arXiv:1807.05031*, 2018.

[173] Levent Sagun, Leon Bottou, and Yann LeCun. Eigenvalues of the hessian in deep learning: Singularity and beyond. *arXiv preprint arXiv:1611.07476*, 2016.

[174] Lloyd N Trefethen and David Bau III. *Numerical linear algebra*, volume 50. Siam, 1997.

[175] Carsen Stringer, Marius Pachitariu, Nicholas Steinmetz, Matteo Carandini, and Kenneth D Harris. High-dimensional geometry of population responses in visual cortex. *Nature*, 571(7765):361–365, 2019.

[176] Arna Ghosh, Arnab Kumar Mondal, Kumar Krishna Agrawal, and Blake Richards. Investigating power laws in deep representation learning. *arXiv preprint arXiv:2202.05808*, 2022.

[177] Guozhang Chen, Franz Scherr, and Wolfgang Maass. Analysis of visual processing capabilities and neural coding strategies of a detailed model for laminar cortical microcircuits in mouse v1. *bioRxiv*, 2021.

[178] Emre O Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019.

[179] Kaushik Roy, Akhilesh Jaiswal, and Priyadarshini Panda. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, 2019.

[180] Benjamin Cramer, Sebastian Billaudelle, Simeon Kanya, Aron Leibfried, Andreas Grübl, Vitali Karasenko, Christian Pehle, Korbinian Schreiber, Yannik Stradmann, Johannes Weis, et al. Surrogate gradients for analog neuromorphic computing. *Proceedings of the National Academy of Sciences*, 119(4), 2022.

[181] Dongsung Huh and Terrence J Sejnowski. Gradient descent for spiking neural networks. *Advances in neural information processing systems*, 31, 2018.

[182] Friedemann Zenke and Surya Ganguli. Superspike: Supervised learning in multilayer spiking neural networks. *Neural computation*, 30(6):1514–1541, 2018.

[183] Wenrui Zhang and Peng Li. Temporal spike sequence learning via backpropagation for deep spiking neural networks. *Advances in Neural Information Processing Systems*, 33:12022–12033, 2020.

[184] Mingqing Xiao, Qingyan Meng, Zongpeng Zhang, Yisen Wang, and Zhouchen Lin. Training feedback spiking neural networks by implicit differentiation on the equilibrium state. *Advances in Neural Information Processing Systems*, 34, 2021.

[185] Nicolas Perez-Nieves and Dan Goodman. Sparse spiking gradient descent. *Advances in Neural Information Processing Systems*, 34, 2021.

[186] Yuhang Li, Yufei Guo, Shanghang Zhang, Shikuang Deng, Yongqing Hai, and Shi Gu. Differentiable spike: Rethinking gradient-descent for training spiking neural networks. *Advances in Neural Information Processing Systems*, 34, 2021.

[187] Nicolas Skatchkovsky, Osvaldo Simeone, and Hyeryung Jang. Learning to time-decode in spiking neural networks through the information bottleneck. *arXiv preprint arXiv:2106.01177*, 2021.

[188] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. {TensorFlow}: A system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

[189] Guillaume Bellec, Darjan Salaj, Anand Subramoney, Robert Legenstein, and Wolfgang Maass. Long short-term memory and learning-to-learn in networks of spiking neurons. *Advances in neural information processing systems*, 31, 2018.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.

- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] To make this easier for the readers, we have referred to the pertinent Figures and Theorem to support our summary of contributions in Introduction.
   (b) Did you describe the limitations of your work? [Yes] Limitations and future work can be found in our Discussion section.
   (c) Did you discuss any potential negative societal impacts of your work? [No] This work contributes to the theoretical understanding of biologically-plausible learning rules for recurrent neural networks. As such, we do not anticipate any direct ethical or societal impact. In the long-term, our work can have impact on related research communities such as neuroscience and deep learning, with societal impact depending on the development of these fields.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] We have carefully read the ethics review guidelines and confirm that this paper conforms to them.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] We state all assumptions precisely when we first presented the Theorem in Results, We further details, discussions and proofs in the appendix.
   (b) Did you include complete proofs of all theoretical results? [Yes] Proofs can be found in Appendix B.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Anonymized code link is provided in Appendix A.3.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] All training details are provided in Appendix A.3.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We tried to provide this information in all applicable figures. This is stated as "Solid lines/shaded regions: mean/standard deviation obtained at the end of training across five independent runs" in the legends of several figures.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Information pertaining to computing resources and simulation time can be found in Appendix A.3.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] For dataset, we cited MNIST as well as representative neuroscience papers for the two neuroscience motivated tasks. We have also cited existing coding assets in Appendix A.3.
   (b) Did you mention the license of the assets? [Yes] We mentioned that in Appendix A.3.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Methods

A.1 Hessian eigenspectrum analysis

As mentioned, we focus on the leading Hessian eigenvalue because it has been used previously [121, 172] and is very feasible for both empirical [118, 173] and theoretical analyses (Theorem 1). The leading Hessian eigenvalue can be computed by performing power iterations on the Hessian vector product without knowing the full Hessian matrix (Algorithm 2 in [118]). To find multiple top eigenvalues, e.g. top 200 eigenvalues, one can use generalized power method via QR decomposition [174]. We focused on the loss' Hessian for recurrent weights but observed similar trend for input weights as well. Stopping procedure for the power iteration were similar to that of [118]; we set the tolerance to $1e-6$, and observed that all iterations in our experiments converged before reaching the maximum iterations allowed, which was set to 500.

Due to the scale-dependence issue of Hessian spectrum [133], we also used scale-independent measures. For instance, we looked at the recently proposed relative flatness measure [6] (Figure 8 in Appendix). We also examined the power-law decay coefficient for the Hessian eigenvalues (Figure 7 in Appendix). We used the code in [175] to fit a power-law distribution to top 200 eigenvalues. We found similar results had we chosen top 50 or 100 eigenvalues instead, and 200 was chosen mainly due to computational load. We note that the link of power-law decay to generalization has also been examined in some recent studies [120, 176, 177].

A.2 Network setup and learning rule implementations

Neuron Model: We consider a discrete-time implementation of a rate-based recurrent neural network (RNN) similar to the form in [136]. The model denotes the internal hidden state as h_t and the observable states, i.e. firing rates, as $f(h_t)$ at time t, and we use ReLU activation for f. The dynamics of those states are governed by

$$h_{t+1} = \alpha h_t + (1-\alpha)\left(W_h f(h_t) + W_x x_t\right), \tag{10}$$

where $\alpha = e^{-dt/\tau_m}$ denotes the leak factor for simulation time step dt and membrane time constant τ_m, W_h denotes the weight of the recurrent synaptic connection, W_x denotes the strength of the input synaptic connection and x_t denotes the external input at time t. we use subscripts to represent indices of neurons and time steps. For instance, $h_{i,t}$ represents the hidden activity h of neuron i at time t. $W_{h,ij}$ represents the $(ij)^{th}$ entry of recurrent weight matrix W_h. Model in Eq. 10 were used for the sequential MNIST and pattern generation tasks.

We mention in passing that the choice of ReLU activation, which has discontinuous first derivative, means that the loss Hessian matrix is not guaranteed to be symmetric. A real matrix that is not symmetric can have complex eigenvalues come in conjugate pairs, and if they were amongst the top eigenvalues, power iterations may not converge. However, all iterations have converged in our experiments as mentioned above. Also, because of potential technical issues resulting from non-symmetric Hessian matrices, we foresee challenges in applying our methodology to spiking neural networks (SNNs), which have discontinuous activation functions. Due to the energy efficiency and biological realism of SNNs [96, 178–187], we believe extending to SNNs is an important future direction.

For the delayed match to sample task, which is a working memory task, it was found in [13] and [11] that units with adaptive threshold as an additional hidden variable can play an important role in the computing capabilities of RNNs. Thus, we implemented adaptive threshold neuron units [164] for that task. In our rate-based implementation, this turns out to be a simple addition of a second hidden variable b_t that represents the dynamic threshold component:

$$\begin{aligned} h_{t+1} &= \alpha h_t + (1-\alpha)\left(W_h f(h_t - b_t) + W_x x_t\right), \\ b_{t+1} &= \beta b_t + (1-\beta) f(h_t - b_t), \end{aligned} \tag{11}$$

where $b_{j,t}$ denotes the dynamic threshold that adapts based on past neuron activity. The decay factor β is given by e^{-dt/τ_b} for simulation time step dt and adaptation time constant τ_b, which is typically chosen on the behavioral task time scale [13].

Network output and loss function:

Readout $\hat{y}$ is defined as

$$\hat{y} = \langle w, f(h_t) \rangle \tag{12}$$

for readout weights w.

We quantify how well the network output $\hat{y}$ matches the desired target y using loss function L, which is defined as

$$L(W_h) = \begin{cases} \frac{1}{2TB} \sum_{i=1}^{B} \sum_{t=1}^{T} \sum_{k=1}^{N_{out}} (\hat{y}_{k,t}^{(i)} - y_{k,t}^{(i)})^2, & \text{for regression tasks} \\ \frac{-1}{TB} \sum_{i=1}^{B} \sum_{t=1}^{T} \sum_{k=1}^{N_{out}} \pi_{k,t}^{(i)} log \hat{\pi}_{k,t}^{(i)}, & \text{for classification tasks} \end{cases} \tag{13}$$

for target output y, task duration T, N_{out} output neurons and batch size B. $\pi_{k,t}$ is the one-hot encoded target and $\hat{\pi}_{k,t} = \text{softmax}_k(\hat{y}_{1,t}, \ldots, \hat{y}_{N_{OUT},t}) = \exp(\hat{y}_{k,t}) / \sum_{k'} \exp(\hat{y}_{k',t})$ is the predicted category probability.

Biological gradient approximations (truncation-based)

The goal of this subsection is to explain where the approximation happens for each of the bio-plausible learning rules. For full details regarding these rules, we encourage to reader to refer to the respective references. We start by writing down the gradient in terms of real-time recurrent learning (RTRL) factorization:

$$\frac{\partial L}{\partial W_{h,ij}} = \sum_{l,t} \frac{\partial L}{\partial h_{l,t}} \frac{\partial h_{l,t}}{\partial W_{h,ij}}, \tag{14}$$

Key problems that RTRL poses to biological plausibility and computational cost reside in the second factor $\frac{\partial h_{l,t}}{\partial W_{h,ij}}$ that arises during the factorization of the gradient (Eq. 14). The factor $\frac{\partial h_{l,t}}{\partial W_{h,ij}}$ keeps track of all recursive dependencies of $h_{l,t}$ on weight $W_{h,ij}$ arising from recurrent connections. These recurrent dependencies can be obtained recursively as follows:

$$\begin{aligned} \frac{\partial h_{l,t}}{\partial W_{h,ij}} &= \frac{\partial h_{j,t}}{\partial W_{h,ij}} + \sum_m \frac{\partial h_{l,t}}{\partial h_{m,t-1}} \frac{\partial h_{m,t-1}}{\partial W_{h,ij}} \\ &= \frac{\partial h_{l,t}}{\partial W_{h,ij}} + \frac{\partial h_{l,t}}{\partial h_{l,t-1}} \frac{\partial h_{l,t-1}}{\partial W_{h,ij}} + \underbrace{\sum_{m \neq l} W_{h,lm} f'(h_{m,t-1}) \frac{\partial h_{m,t-1}}{\partial W_{h,ij}}}_{\text{depends on all weights } W_{h,lm}}. \end{aligned} \tag{15}$$

Thus, the factor $\frac{\partial h_{l,t}}{\partial W_{h,ij}}$ and it poses a serious problem for biological plausibility: it involves *nonlocal* terms that should be inaccessible to neural circuits, i.e. that knowledge of all other weights in the network is required in order to update the weight $W_{h,ij}$.

RFLO [12] (labeled as "RF Three-factor") and *symmetric e-prop* [11] (labeled as "Three-factor") seek to address this by truncating the expensive nonlocal term in Eq. 15 so that the updates to weight $W_{h,ij}$ would only depend on pre- and post-synaptic activity:

$$\widehat{\frac{\partial h_{l,t}}{\partial W_{h,ij}}} = \begin{cases} \frac{\partial h_{i,t}}{\partial W_{h,ij}} + \frac{\partial h_{i,t}}{\partial h_{i,t-1}} \widehat{\frac{\partial h_{i,t-1}}{\partial W_{h,ij}}}, & l = i \\ 0, & l \neq i \end{cases} \tag{16}$$

which results in a much simpler factor than the triple tensor in Eq. 15.

After the truncation, RFLO and e-prop implement:

$$\widehat{\frac{\partial L}{\partial W_{h,ij}}} = \sum_t \frac{\partial L}{\partial h_{i,t}} \widehat{\frac{\partial h_{i,t}}{\partial W_{h,ij}}}, \tag{17}$$

$$\widehat{\frac{\partial h_{i,t}}{\partial W_{h,ij}}} = \frac{\partial h_{i,t}}{\partial W_{h,ij}} + \frac{\partial h_{i,t}}{\partial h_{i,t-1}} \widehat{\frac{\partial h_{i,t-1}}{\partial W_{h,ij}}}. \tag{18}$$

The main difference between symmetric e-prop and RFLO implementation is that symmetric feedback is used for symmetric e-prop, i.e. output weight w is used as the feedback weight for the $\frac{\partial E}{\partial h}$, whereas RFLO uses fixed random feedback weights [75] for greater biological plausibility. We note in passing that the authors of e-prop have tested their formulation with fixed random feedback weights

as well. *MDGL* [13] also truncates RTRL, but it restores some of the non-local dependencies – those within one connection step — that could potentially be communicated via mechanisms similar to the abundant cell-type-specific local modulatory signaling unveiled by recent transcriptomics data [89, 90]. With that, the expensive memory trace term in Eq. 15 becomes

$$\frac{\partial h_{l,t}}{\partial W_{h,ij}} \approx \begin{cases} W_{h,li} f'(h_{i,t-1}) \widehat{\frac{\partial h_{i,t-1}}{\partial W_{h,ij}}}, & i \neq l \\ \frac{\partial h_{i,t}}{\partial W_{h,ij}} + \frac{\partial h_{i,t}}{\partial h_{i,t-1}} \widehat{\frac{\partial h_{i,t-1}}{\partial w_{ij}}}, & i = l \end{cases} \tag{19}$$

MDGL introduces one additional approximation: replace $W_{h,li}$ with type-specific weights W_{ab} to mimick the cell-type-specific nature of local modulatory signaling (for cell i in group a and cell j in group b, where $a, b \in C$ for a total of C cell groups). For simplicity, we just use $W_{ab} = W_{li}$, i.e. without cell-type approximation. This results in overall MDGL implementation as

$$\widehat{\frac{\partial L}{\partial W_{h,ij}}} = \sum_t \frac{\partial L}{\partial h_{i,t}} \widehat{\frac{\partial h_{i,t}}{\partial W_{h,ij}}} + \widehat{\frac{\partial h_{i,t-1}}{\partial W_{h,ij}}} \sum_l W_{h,li} \frac{\partial L}{\partial h_{l,t}}, \tag{20}$$

Interpretation of the above update rule in terms of biological processes can be found in the MDGL paper [13].

We note that input, recurrent and output weights were all being trained. This section illustrates the approximate gradient for updating recurrent weights W_h, and similar expressions can be obtained for updating input weights W_x. The approximations, however, do not apply to output weights, as the gradient for that does not violate the aforementioned issue of nonlocality (Eq. 15).

A.3 Simulation details

We used TensorFlow [188] version 1.14 and based it on top of [189]. We modified the code for rate-based neurons (Eq. 10 and 11). [1] We used the code in [175] for the power-law analysis (Figure 7 in Appendix). SGD optimizer was used to study the effect of gradient approximation in isolation without the complication of additional factors, as Adam optimizer with adaptive learning rate could convolute our matching step length experiments in Figure 4. That said, we verified that the curvature convergence behavior is also observed for Adam optimizer (Figure 11 in Appendix. Learning rates were optimized by picking within $\{3e-4, 1e-3, 3e-3, 1e-2, 3e-2, 1e-1\}$ for each algorithm. For the sequential MNIST task, we explored batch sizes within $\{64, 256, 1024\}$. Trainings were stopped when both the loss and leading Hessian eigenvalue stabilized. As stated, we repeated runs with different random initialization to quantify uncertainty and weights were initialized similarly as in [12].

Simulations were completed on a computer server with x2 20-core Intel(R) Xeon(R) CPU E5-2698 v4 at 2.20GHz. The average time to complete one run of sequential MNIST, pattern generation and delayed match to sample tasks in Figure 3 were approximately 2 hours, 1 hour and 1 hour, respectively. Since the computation of second order gradient becomes prohibitively expensive as sequence length T becomes large, all tasks involved no more than 50 time steps. For instance, this was achieved for the sequential MNIST task using the row-by-row implementation. Using fewer time steps, however, should not affect the general trend as the gradient truncation effects were still significant. Because of the use of fewer steps, we dropped the leak factor α in Eq. 10 (i.e. set $\alpha = 0$).

For the matching step length experiments (Figure 4), we simply obtained $\rho = \frac{\vec{\bar{g}}^T \vec{g}}{\vec{\bar{g}}^T \vec{g}}$ for the three-factor learning rule and scaled BPTT updates by that amount. For scheduled learning rate experiments (Figure 5), the additional hyperparameters include initial learning rate, decay percentage and decay frequency. We used an initial learning rate that is three times the uniform rate (used in other figures) and decay the learning rate by 80% every X iterations, where X is roughly the total number of training iterations (used in other figures) divided by 30. Since the point of that figure was to show that learning rate scheduling could lead to flatter minima than using a fixed learning rate, we did not search extensively across these additional hyperparameters as the first set of hyperparameters we tried was enough to demonstrate that point.

[1] Our code is available: `https://anonymous.4open.science/r/BiolHessRNN-4DA5/README.md`.

For the pattern generation task, our network consisted of $N = 30$ neurons described in Eq. 10. Input to this network was provided by a random Gaussian input ($N_{in} = 1$). The fixed target signal had a duration of 50 steps and given by the sum of four sinusoids, with fixed period of 10, 40, 70 and 100 steps. For learning, we used mean squared loss function. Training for this task used full batch. For testing, we perturbed the input with additive zero-mean Gaussian noise (with σ picked uniformly between 0 and 0.2 across runs), to mimick the situation where the agent has to faithfully produce the desired pattern even under perturbations. Unlike the other tasks, this task measures accuracy by mean squared error, for which the lower the better. To maintain the convention of higher generalization gap being worse, the generalization gap for this task is computed by test error minus train error.

For the delayed match to sample task, our network consisted of $N = 100$ neurons, which include 50 neurons with (Eq. 11) and 50 neurons without (Eq. 10) threshold adaptation. The task involved the presentation of two sequential cues, each taking on a binary value, lasting 2 steps and separated by a delay of 16 steps. Input to this network was provided by $N_{in} = 2$ neurons. The first (resp. second) input neuron sends a value of 1 when the presented cue takes on a value of 1 (resp. cue 0), and 0 otherwise. The network was trained to output 1 (resp. 0) when the two cues have matching (resp. non-matching) values. For learning, we used cross-entropy loss function and the target corresponding to the correct output was given at the end of the trial. Training for this task used full batch. For testing, we tested on increased delay, with the period picked uniformly between the training delay period and twice the training delay period, to mimick situations where the animal has to hold the memory longer than it did during the learning phase.

For the sequential MNIST task [137], our network consisted of $N = 128$ neurons described in Eq. 10. Input to this network was provided by $N_{in} = 28$ units that represented the grey-scaled value of a single row, totalling 28 steps and the network prediction was made at the last step. For learning, we used the cross-entropy loss function and the target corresponding to the correct output was given at the end of the trial. For testing, we used the existing MNIST test set [137] with additive zero-mean Gaussian input noise. As mentioned, this task did not train with full batch but we found the trend to hold across different batch sizes.

Finally, we note that comparisons BPTT and approximate rules are done at comparable training accuracies for the pattern generation and delayed match to sample tasks. For the sequential MNIST task, the three-factor rule achieves only around 70% training accuracy, and even then the curvature convergence behavior does not predict the training accuracy. To see this, while three-factor theory (blue in Fig 4), which corresponds to BPTT with reduced step length, achieves an accuracy of $> 95\%$ but still attains similar curvature convergence to that of the three-factor rule.

B Theorem 1

B.1 Proof of Theorem 1

Proof. First, we note that the Jacobian of the dynamical system for BPTT update (Eq. 7) is simply the loss Hessian scaled by $-\eta$. This implies that

$$|\lambda_1^J| = \eta_B |\lambda_1^H|, \tag{21}$$

where we remind the reader that $\lambda_1^J \in \mathbb{R}$ (resp. $\widehat{\lambda_1^J} \in \mathbb{R}$) is the leading eigenvalue for BPTT (resp. approximate rule) Jacobian; $\lambda_1^H \in \mathbb{R}$ is the leading eigenvalue of the loss' Hessian matrix.

Second, with the assumption of a single output $\hat{y}$, loss presented only at the last time step T and stochastic gradient descent (updates on a single data example as opposed to batch updates), least squares loss in Eq. 3 can be simplified to:

$$L(W_h) = \frac{1}{2}(\hat{y} - y)^2, \tag{22}$$

resulting in the following difference equations for discrete dynamical systems defined by BPTT (Eq. 7) and approximate rule (Eq. 8):

$$\Delta W_h|_{\text{BPTT}} = F(W_h) = -\eta_B(\hat{y} - y)\nabla\hat{y}, \tag{23}$$

$$\Delta W_h|_{\text{approximate}} = \hat{F}(W_h) = -\eta_e(\hat{y} - y)\tilde{\nabla}\hat{y}, \tag{24}$$

where we remind the reader that $\tilde{\nabla}$ is the notation for an approximate gradient.

We then compute the Jacobian of the difference equations above:

$$J = -\eta_B \left(\nabla\hat{y}\nabla\hat{y}^T + (\hat{y} - y)\nabla^2\hat{y}\right) \quad \text{(for BPTT)} \tag{25}$$

$$\widehat{J} = -\eta_e \left(\nabla\hat{y}\tilde{\nabla}\hat{y}^T + (\hat{y} - y)\nabla\tilde{\nabla}\hat{y}\right) \quad \text{(for approximate rule).} \tag{26}$$

In the limit of zero error $(y - \hat{y}) = 0$, i.e. close to an optimum, the term involving $(y - \hat{y})$ becomes negligible. That simplifies the Jacobian to

$$\begin{aligned} J &\approx -\eta_B \nabla\hat{y}\nabla\hat{y}^T \\ \widehat{J} &\approx -\eta_e \nabla\hat{y}\tilde{\nabla}\hat{y}^T. \end{aligned} \tag{27}$$

In this case, J and $\widehat{J}$ are *rank-1* matrices. A rank-1 square matrix has only one nonzero eigenvalue, and by inspection, that one eigenvalue is

$$\text{For } J: |\lambda_1^J(W_B^*)| \quad = \eta_B \nabla\hat{y}^T\nabla\hat{y}\big|_{W_B^*} \tag{28}$$

$$\rightarrow |\lambda^H| \overset{(21)}{=} |\lambda_1^J|/\eta_B = \nabla\hat{y}^T\nabla\hat{y} \tag{29}$$

$$\begin{aligned} \text{For } \widehat{J}: \quad |\widehat{\lambda_1^J}(W_e^*)| &= \eta_e |\tilde{\nabla}\hat{y}^T\nabla\hat{y}|\Big|_{W_e^*} \\ &\overset{(a)}{=} |\eta_e \rho \nabla\hat{y}^T\nabla\hat{y}|\big|_{W_e^*} \\ &\overset{(29)}{=} |\rho\eta_e \lambda_1^H(W_e^*)|, \end{aligned} \tag{30}$$

where equality (a) is explained as follows. We first remind the reader that ρ is defined such that $\vec{\tilde{g}} = \rho\vec{g} + \vec{e}$ (Eq. 6). For the case of a scalar output $\hat{y}$, $\vec{\tilde{g}} = \frac{\partial L}{\partial \hat{y}}\tilde{\nabla}\hat{y}$ and $\vec{g} = \frac{\partial L}{\partial \hat{y}}\nabla\hat{y}$. So if we divide both sides of $\vec{\tilde{g}} = \rho\vec{g} + \vec{e}$ by $\frac{\partial L}{\partial \hat{y}}$ we get $\tilde{\nabla}\hat{y} = \rho\nabla\hat{y} + \vec{e}/\frac{\partial L}{\partial \hat{y}}$. Since we have $\vec{e} \perp \vec{g}$ by definition, then $\vec{e}^\top\nabla\hat{y} = 0$ because $\vec{g}$ is just a scaled $\nabla\hat{y}$ when the output is a scalar. This leads to $\tilde{\nabla}\hat{y}^T\nabla\hat{y} = (\rho\nabla\hat{y} + \vec{e}/\frac{\partial L}{\partial \hat{y}})^\top\nabla\hat{y} = \rho\nabla\hat{y}^T\nabla\hat{y}$.

Since we assume the gradient descent dynamical system converges to an optimum, this corresponds to an asymptotic stable fixed point. Hence, $|\lambda_1^J| < 1$ and $|\widehat{\lambda_1^J}| < 1$, which implies:

$$|\lambda_1^J(W_B^*)| < 1 \overset{(21)}{\rightarrow} \eta_B|\lambda_1^H(W_B^*)| < 1 \rightarrow |\lambda_1^H(W_B^*)| < \frac{1}{\eta_B} \tag{31}$$

$$|\widehat{\lambda_1^J}(W_e^*)| < 1 \overset{(30)}{\rightarrow} |\rho\eta_e\lambda_1^H(W_e^*)| < 1 \rightarrow |\lambda_1^H(W_e^*)| < \frac{1}{|\rho|\eta_e}. \tag{32}$$

□

B.2 Discussion on tightness of the bound

Following the derivation, it is clear that the tightness of the bound will depend on how close the magnitude of leading Jacobian eigenvalues are to 1 upon convergence. That is related to the distribution of minima flatness along the loss landscape, which impacts the probability of a rule converging to a minima with flatness in a certain range. Such distribution is likely problem dependent. In the extreme scenario where the loss is convex, there is just one minimum, so the question of minima preference becomes irrelevant.

B.3 Discussion on generality of Theorem 1

The proof above examines a special case where the Jacobian of weight update equations becomes rank-1. We remind the reader that for the case of multivariate loss, higher rank case or batch updates, we would not arrive at the rank-1 Jacobian step (Eq. 27). For many tasks considered in neuroscience, the rank 1 case can apply, which explains the validity of Theorem 1. We also note in passing that for the case of stochastic gradient descent, the Hessian Jacobian correspondence (Eq. 21) would point to loss' Hessian matrix evaluated on a single example, which could reflect the robustness against perturbing that particular example. Moreover, simulation results show our conclusion holds in higher rank cases (Figure 4). The challenge of generalizing the proof to higher Jacobian rank case is that we are no longer guaranteed that the leading eigenvectors of BPTT Jacobian coincide with the leading eigenvectors of approximate rule Jacobian. Thus, it becomes much harder to relate $|\widehat{\lambda_1^J}|$ and $|\lambda_1^J|$. Rather than providing further proof, we provide an intuition for why our conclusion — where the convergence behavior between rules differs by their step length along the gradient direction — can hold in higher rank case under Assumption 1.

Assumption 1. *Approximation error vector $\vec{e}$ (but not $\vec{g}$) lies orthogonal to the subspace spanned by the leading Hessian eigenvectors. Here, leading Hessian eigenvectors refer to the eigenvectors corresponding to the outlier Hessian eigenvalues, given the well known observation that there exists only a few large (outlier) eigenvalues and the rest are near zero [116, 117, 173]).*

The ramification of Assumption 1 is that $\vec{e}$ will lie in the subspace spanned by eigenvectors corresponding to tiny eigenvalues, making $H\vec{e}$ tiny. In the extreme scenario where $\vec{e}$ lies in the null space of H, $H\vec{e}$ would be 0. We verify this assumption numerically in Figure 10. We saw from the proof above that this assumption is automatically satisfied in the rank-1 Jacobian case. We remark that this assumption should not hold for stochastic gradient noise (SGN), as SGN covariance matrix is well aligned with the Hessian matrix near a minima [122]. This could be why $\vec{e}$, unlike stochastic gradient noise, does not seem to be contributing much to escaping narrow minima.

We consider the case of small enough weight updates such that the loss surface can be approximated using second order Taylor expansion. Thus, the loss change after one update becomes:

$$\begin{aligned}\Delta L &\approx \Delta W^T \vec{g} + \frac{1}{2}\Delta W^T H \Delta W \\ &= -\eta_B \vec{g}^T \vec{g} + \frac{1}{2}\eta_B^2 \vec{g}^T H \vec{g}, \quad \text{(for exact rule)} \qquad (33)\\ \widehat{\Delta L} &\approx \widehat{\Delta W}^T \vec{g} + \frac{1}{2}\widehat{\Delta W}^T H \widehat{\Delta W} \\ &= -\eta_e \vec{\hat{g}}^T \vec{g} + \frac{1}{2}\eta_e^2 \vec{\hat{g}}^T H \vec{\hat{g}}. \quad \text{(for approximate rule)} \qquad (34)\end{aligned}$$

We next focus on the first and second Taylor term (T_1 and T_2) for the exact rule as well as the terms ($\widehat{T_1}$ and $\widehat{T_2}$) for the approximate rule:

$$T_1 := \eta_B \vec{g}^T \vec{g}, \widehat{T_1} := \eta_e \vec{\hat{g}}^T \vec{g}, T_2 := \frac{1}{2}\eta_B^2 \vec{g}^T H \vec{g}, \widehat{T_2} := \frac{1}{2}\eta_B^2 \vec{\hat{g}}^T H \vec{\hat{g}},$$

and we note that first and second Taylor terms can determine how likely the update will be trapped in a local minima:

$$\begin{aligned}
&\Delta L < 0 \text{ (for going further down)} \quad \rightarrow T_1 > T_2 \\
&\Delta L > 0 \text{ (restricts convergence)} \quad \rightarrow T_1 < T_2 \\
&\widehat{\Delta L} < 0 \text{ (for going further down)} \quad \rightarrow \widehat{T_1} > \widehat{T_2} \\
&\widehat{\Delta L} > 0 \text{ (restricts convergence)} \quad \rightarrow \widehat{T_1} < \widehat{T_2}.
\end{aligned} \tag{35}$$

Given their central role in determining convergence, we compare these terms between exact rule and approximate rule. For the first Taylor term (T1), it is easy to see that:

$$\vec{\hat{g}}^T \vec{g} = \rho \vec{g}^T \vec{g}.$$

For the second Taylor term (T2) and if H is symmetric:

$$\begin{aligned}
\vec{\hat{g}}^T H \vec{\hat{g}} &= \rho^2 \vec{g}^T H \vec{g} + 2\rho \vec{g}^T \underbrace{H\vec{e}}_{\approx 0} + \rho^2 \vec{e}^T \underbrace{H\vec{e}}_{\approx 0, \text{ Assumption 1}} \\
&\approx \rho^2 \vec{g}^T H \vec{g}.
\end{aligned} \tag{36}$$

To match the convergence behavior between exact and approximate rules (Eq. 35) for a (locally) second order loss surface, we can make (T_1, T_2) approximately equal to $(\widehat{T_1}, \widehat{T_2})$ by setting $\eta_B = \rho \eta_e$, which predicts our numerical results (Figure 4). We note that if Assumption 1 is not satisfied, then the above may not hold. Whether we would still need the negligible training error assumption depends on how well Assumption 1 can be satisfied without being near an optimum.

C Additional Simulations

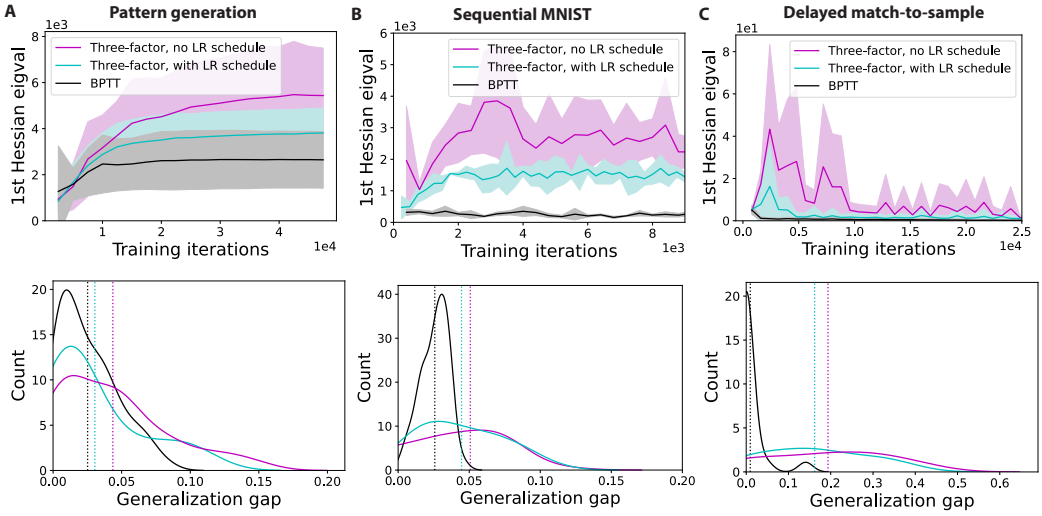

Figure 5: **Learning rate modulation as a possible remedy of the problem**. We increased the learning rate in the beginning of training to prevent the three-factor rule from stabilizing in sharp minima prematurely, followed by a gradual decay to prevent instability. Top panels show this strategy helps to reduce the curvature of the converged solution. Bottom panels show this leads to a slight improvement in generalization gap (vertical lines denote distribution mean). However, it is important to note that this strategy does not correct the problem; the gap still exists compared to BPTT, suggesting room for further research. Plotting conventions follow that of the previous figures.

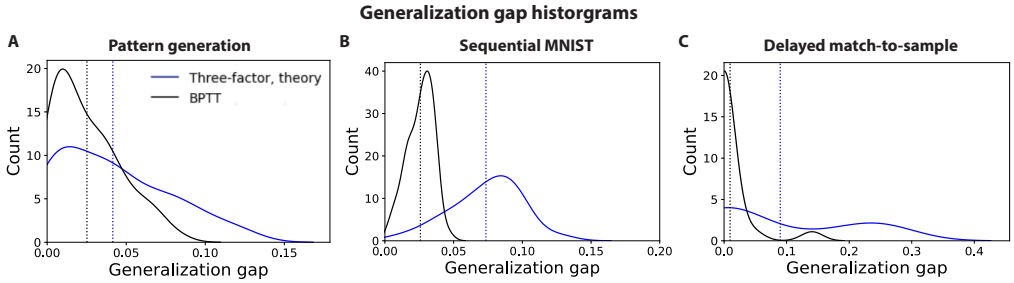

Figure 6: **Modified BPTT (three-factor, theory) resulted in worse and more variable generalization performance**. Here, we follow the convention of previous generalization gap histogram plots and investigated the generalization performance of modified BPTT (three-factor, theory) in Figure 4).

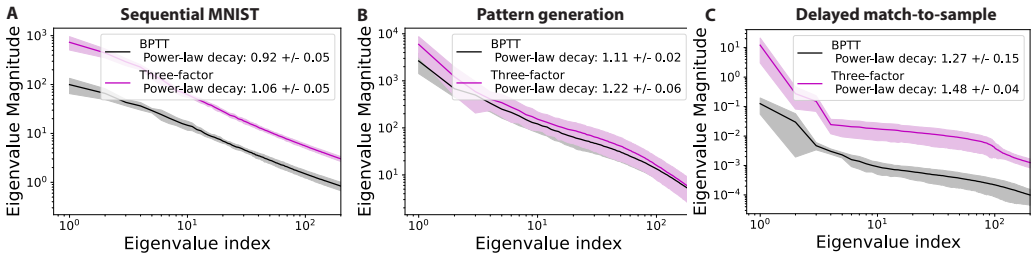

Figure 7: **Loss' Hessian eigenspectrum for the three-factor rule exhibits significantly steeper power-law decay compared to that of BPTT**. We fit a power-law function to the top 200 eigenvalues at the end of training and measure the decay parameter. Similar results were obtained if we fit to the top 50 or 100 eigenvalues. Solid lines/shaded regions: mean/standard deviation of eigenspectrum obtained at the end of training across five independent runs.

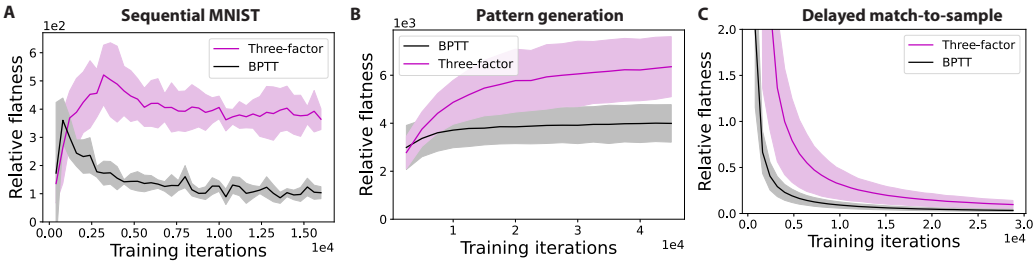

Figure 8: **Curvature preference behavior also corroborated using relative flatness measure [6]**. Here, the trend is consistent with that of Figure 3. Note that the relative flatness measure can be computationally intensive for recurrent weights, so we computed it for readout weights. Plotting conventions follow that of previous figures.

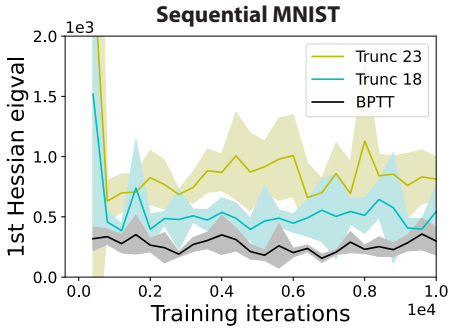

Figure 9: **Approaching high curvature regions seems to be a shared problem for temporal truncations of the gradient**. We repeat the analysis in Figure 3 for truncated BPTT (TBPTT). Here, "Trunc X" means X time steps are truncated during the gradient calculation. We observe TBPTT tend to converge to high curvature regions. Plotting conventions follow that of previous figures.

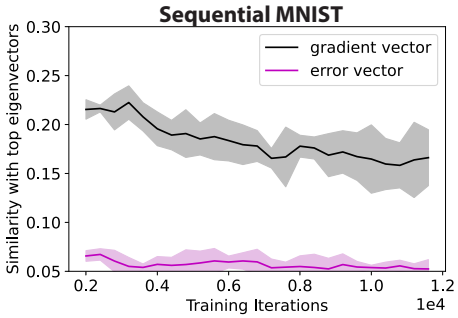

Figure 10: **Truncation error vector (compared to the gradient) is significantly less aligned with the top Hessian eigenvector subspace**. Following [172], we compute the cosine similarity with a top Hessian eigenvector (averaged over the top 5 eigenvectors). The absolute value of the cosine similarity was taken. We observe weak alignment of the approximation error vector $\vec{e}$ with the leading Hessian eigenvectors. Similar trends were attained had we averaged over the leading 10 or 20 eigenvectors. Plotting conventions follow that of previous figures.

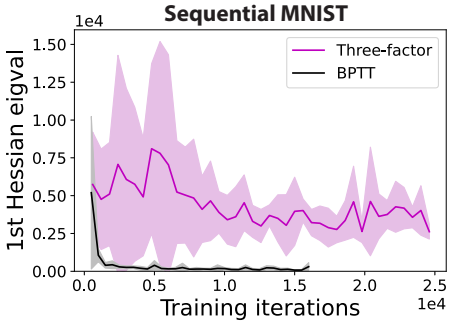

Figure 11: **Curvature convergence behavior also holds for Adam optimizer**. As explained in Methods, we use SGD optimizer due to confounding factors in Adam optimizer that could convolute our matching step length analysis in Figure 4. We observe similar trends as in Figure 3. Plotting conventions follow that of previous figures.

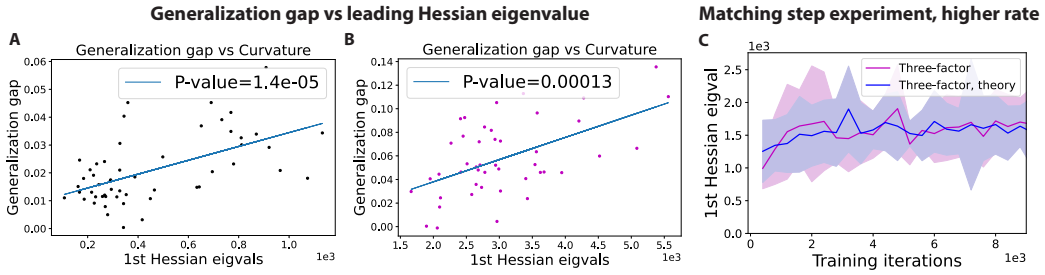

Figure 12: We repeat the generalization gap vs leading Hessian eigenvalue scatter plot in Figure 2 with the learning rule fixed for A) BPTT and B) the three-factor rule. As expected, significant correlation between generalization gap and leading Hessian eigenvalue is observed. Note, unlike Figure 2, where the hyperparameters are fixed for each rule (tuned using the procedure in Appendix A.3), the learning rate is varied here in order to get a wide enough curvature range to observe the correlation. C) Matching step experiment in Figure 4 repeated with a learning rate three times of that in Figure, and the observation agrees with that in Figure 4. Plotting convention follows that of previous figures.

Learning	Leading Hessian eigenvalue	Generalization gap
Three-factor	2550 ± 490	0.5 ± 0.3
BPTT, early stopping	316 ± 84	0.2 ± 0.1

Table 1: BPTT early stopped to match the test accuracy of e-prop for the sequential MNIST task. Higher generalization gap and leading Hessian eigenvalue is again observed as expected. Each rule is repeated for five runs with different random weight initialization each time.