META-LEARNED STATEFUL PLASTICITY: LOCAL RULES FOR ONLINE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Biology motivates learning rules that modify the connection between neurons using only local information available at the synapse. While hand-designed local plasticity rules are interpretable, they are not generally optimized for task objectives. We present Meta-Learned Stateful Plasticity (MLSP), a framework for discovering local rules in online classification settings. The rules are parameterized by multilayer perceptrons (MLPs) that are shared across synapses and take in only locally available information such as pre- and postsynaptic activations, the current connection weight, and a broadcast reward signal, while also updating a writable neuron-local *plasticity state*. We apply MLSP to the hidden weight matrices of a feedforward network whose readout weights learn with the delta rule. The meta-trained rules outperform all tested classical rules, generalize to task families not used in meta-training, and are competitive with, though generally below, online backpropagation. Rules meta-trained with plasticity states reach higher average accuracy than rules without them, and we present evidence that the states track progress through the episode at the layer level. Finally, we fit polynomials of local variables to rules meta-trained without plasticity states. Networks trained with these polynomial rules retain much of the advantage over learning with the delta rule alone, but the fits do not reduce to any classical rule. This indicates that effective local learning rules extend beyond the hand-designed forms typically considered.

1 INTRODUCTION AND CONTRIBUTION

Learning in the brain is likely implemented by updates to the synaptic weights between neurons, using information available locally at each connection together with broadcast modulatory signals. Most proposed plasticity rules are specified in advance, either postulated from general principles, such as Hebb's proposal that coactive neurons strengthen their connection (Hebb, 1949), or designed for a specific computation, such as Oja's rule, which extracts the first principal component of the inputs (Oja, 1982). While these plasticity rules are interpretable, their hand-designed forms may be too restrictive to support learning on difficult tasks. Three-factor rules (Izhikevich, 2007; Frémaux et al., 2010) can connect local plasticity to behavioral outcomes by combining pre- and postsynaptic activity with a reward signal broadcast across the network, but how these signals determine the synaptic update remains hand-designed. Backpropagation derives synaptic updates from precise gradients of a task objective, but standard implementations rely on neuron-specific errors carried by backward weights matched to the forward connections, a requirement known as weight transport (Grossberg, 1987). Dense symmetric feedback connections are not observed in biology, making them an unlikely general mechanism for synaptic credit assignment. Several biologically motivated models have proposed ways of approximating backpropagation while avoiding the weight transport problem (Payeur et al., 2021; Sacramento et al., 2018; Guerguiev et al., 2017), but whether and how these mechanisms are implemented in the brain remains unresolved.

Instead of hand-designing a rule or approximating backpropagation, we ask whether the plasticity rule itself can be learned. We use meta-learning to discover a local plasticity rule that supports online learning of classification tasks, and investigate whether the learned rule can be reduced to a simpler, interpretable form and which local signals it uses. We introduce Meta-Learned Stateful Plasticity (MLSP), a framework for meta-learning a local plasticity rule that updates *all* the hidden weight matrices of a feedforward classifier, which we call the task network. We search over a broad class

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

of local plasticity rules without prescribing the form of the update in advance. The plasticity rule g_θ is parameterized by an MLP with parameters θ, which we call the rule MLP. The same rule MLP is applied at each connection between hidden layers, and its inputs include the pre- and postsynaptic activations, the current weight and a broadcast binary reward that signals whether the prediction was correct. Building on evidence that neurons can maintain local variables that influence plasticity and retain information about their history (Sehgal et al., 2013; Abraham & Bear, 1996), we also allow g_θ to read and update one writable scalar plasticity state, ξ, per hidden neuron. The task network's linear readout instead learns with a supervised delta rule.

Our main contributions are as follows:

- We introduce MLSP, a framework for meta-learning a task-optimized local plasticity rule shared across connections between hidden layers in the task network. Once meta-trained, the plasticity rule uses only connection- and neuron-local information together with a broadcast reward. No gradients or neuron-specific feedback reach the hidden layers.
- We show that MLSP rules enable newly initialized task networks to learn online, outperform the classical Hebb, Oja, BCM and three-factor Hebb rules, and are competitive with, though generally below, online backpropagation. Meta-trained on permuted MNIST (LeCun et al., 1998) and the Random Hierarchy Model (Cagnetta et al., 2024), the rules also transfer to Fashion-MNIST (Xiao et al., 2017) and Omniglot (Lake et al., 2015), neither of which contributes to the meta-training objective.
- We investigate how the learned plasticity rules use the information available to them. Interventions on the plasticity states, together with a step-index control in which the plasticity rule receives the episode step directly, suggest that the plasticity states are used mainly to represent progress through the plasticity episode. We also find that online learning performance depends little on the reward for the task families studied, so the hidden-layer rule is effectively unsupervised and label information enters mainly through the delta-rule readout, while postsynaptic activations and the current connection weight are particularly important for meta-learning an effective plasticity rule.

2 Related work

Meta-learning local plasticity rules. Confavreux et al. (2020) meta-learned the coefficients of a polynomial expansion in pre- and postsynaptic activations and weights to rediscover classical local rules, including in rate networks, whereas Confavreux et al. (2023) used simulation-based inference to find families of polynomial and MLP plasticity rules in recurrent spiking networks. These works aimed to recover known rules for simple tasks or to maintain biologically plausible network dynamics. We pursue the complementary goal of discovering local rules that improve task performance during online learning. Other studies have meta-learned plasticity, feedback, or changing local variables for memory and online credit assignment (Miconi et al., 2018; Lindsey & Litwin-Kumar, 2020; Tyulmankov et al., 2022; Sandler et al., 2021). Meta-learned unsupervised update rules (Metz et al., 2019) and recurrent learning algorithms (Kirsch & Schmidhuber, 2021) have also generalized to datasets not seen during meta-training. Shervani-Tabar & Rosenbaum (2023) meta-learned combinations of plasticity terms that receive neuron-specific errors through random feedback pathways. Metz et al. (2019) meta-learn a shared neuron-local update network and Sandler et al. (2021) shared rules with per-neuron states, but both propagate learned backward messages. Najarro & Risi (2020) evolved synapse-specific Hebbian rule coefficients. MLSP instead meta-learns one shared plasticity rule without neuron-specific feedback for single-sample online learning over a short horizon, and we analyze which rule inputs and plasticity states it uses. Unlike learned optimizers (Andrychowicz et al., 2016; Maheswaranathan et al., 2021), the rule receives no gradient during online learning.

Biologically plausible approximations to backpropagation. Examples include fixed random feedback (Lillicrap et al., 2016), direct feedback alignment (Nøkland, 2016), learned or estimated feedback pathways (Akrout et al., 2019; Kunin et al., 2020), local targets (Lee et al., 2015), burst-dependent updates (Payeur et al., 2021), and e-prop (Bellec et al., 2020). These methods still provide neuron-specific error information. In MLSP, output errors are not propagated to the hidden layers during online learning. The plasticity rule instead uses only local information recorded from a forward pass, and a binary reward.

3 META-LEARNED STATEFUL LOCAL PLASTICITY

MLSP separates learning into two nested processes. At the faster level, a newly initialized task network learns a single classification task from 512 labeled samples presented one at a time. We refer to this sequence as a plasticity episode (Appendix A collects our terms). After each prediction, the correct label is used to update the linear readout (a delta rule), and to compute a binary reward that is broadcast to the local plasticity rule. The plasticity rule uses these inputs to update the task-network hidden weight matrices and plasticity states, while its own parameters remain fixed throughout the episode. At the slower level, meta-training adjusts θ across many plasticity episodes. After each episode, the mean prediction loss over its 512 episode steps is differentiated through the full plasticity episode to update θ, and the task network is newly initialized for the next task. Backpropagation is therefore used to discover the local plasticity rule, but not to update the hidden weight matrices during online learning.

We use task family to refer to a distribution of related classification problems, and task or task instance to refer to one particular problem sampled from that distribution. We meta-train MLSP on two task families, permuted MNIST (pMNIST), in which each task uses a new pixel permutation (LeCun et al., 1998; Goodfellow et al., 2014), and the Random Hierarchy Model (RHM), in which each task is generated from a new hierarchical structure (Cagnetta et al., 2024). Sampling a new task and a newly initialized task network in each plasticity episode requires meta-training to discover a reusable learning strategy rather than a solution to one problem. After meta-training, we fix the plasticity rule and evaluate it on new pMNIST permutations and RHM hierarchies, and test its transfer to Fashion-MNIST (Xiao et al., 2017), with a new pixel permutation in each episode, and to Omniglot tasks that each use 10 of the 659 evaluation-split characters (Lake et al., 2015), neither of which contributes to the meta-training objective.

3.1 ARCHITECTURE

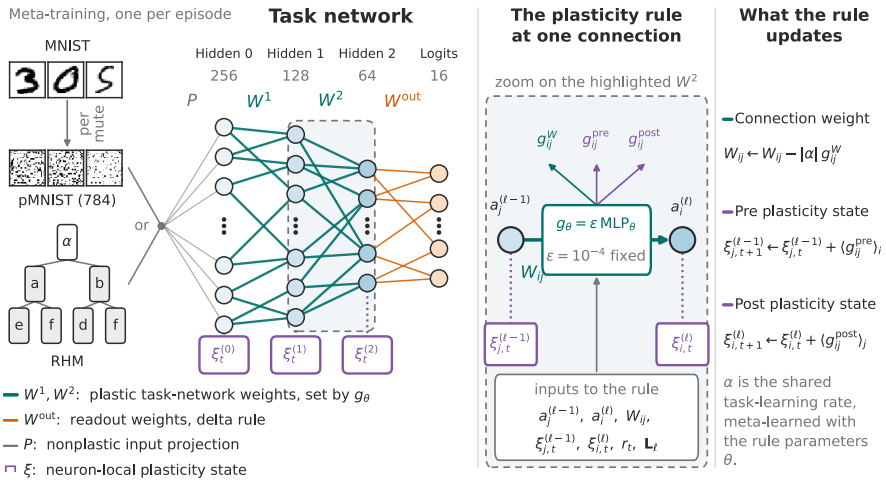

Figure 1: MLSP architecture and local updates. In meta-training, each episode draws a pMNIST or RHM task. In the task network, a nonplastic random input projection P is redrawn each episode. The hidden weight matrices $W^{(1)}$ and $W^{(2)}$ share the rule g_θ, and the readout W^{out} uses a supervised delta rule. Hidden neurons carry plasticity states $\xi_t^{(\ell)}$, which are not forward signals. At one connection, the rule's inputs are the pre- and postsynaptic activations, the weight, the two plasticity states, the binary reward and a one-hot layer indicator. The rule updates the weight and the two plasticity states, and each neuron averages its state updates over its connections.

The task network shown in Figure 1 has a nonplastic random $784 \to 256$ input projection, two hidden weight matrices $W^{(1)}$ and $W^{(2)}$ of sizes $256 \to 128$ and $128 \to 64$, and $64 \to 16$ readout weights. It has no biases. The three hidden layers ($\ell = 0, 1, 2$) use ReLU nonlinearities, and a linear readout followed by softmax gives class-probability predictions. The input projection is redrawn at the beginning of each episode and remains fixed throughout it, discouraging solutions that rely on

a particular input structure. Each hidden neuron carries one writable scalar plasticity state, which does not directly enter the feedforward computation.

For a connection $(j \to i)$ in $W^{(\ell)}$ from layer $\ell - 1$ to layer ℓ, the rule MLP input for episode step t is

$$\phi^{\ell}_{ij,t} = [a^{(\ell-1)}_j, a^{(\ell)}_i, W^{\ell}_{ij}, \xi^{(\ell-1)}_j, \xi^{(\ell)}_i, r, \mathbf{L}^{(\ell-1)}]_t, \qquad r_t = \begin{cases} 1, & \text{if } \arg\max_k o_{t,k} = y_t, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Here $a^{(\ell-1)}_j$ and $a^{(\ell)}_i$ are pre- and postsynaptic activations from the current forward pass, W^{ℓ}_{ij} is the current weight, and $\xi^{(\ell-1)}_j$ and $\xi^{(\ell)}_i$ are the plasticity states of two neurons. The reward r_t is broadcast identically to all connections in $W^{(1)}$ and $W^{(2)}$, and the one-hot layer indicator $\mathbf{L}^{(\ell-1)}$ identifies the presynaptic layer.

The plasticity rule g_θ is parameterized by a single network, MLP_θ, applied independently at every connection in $W^{(1)}$ and $W^{(2)}$. It outputs one weight-update proposal and two plasticity-state-update proposals as

$$(g^{W,\ell}_{ij}, g^{\mathrm{pre},\ell}_{ij}, g^{\mathrm{post},\ell}_{ij})_t = \varepsilon\, \mathrm{MLP}_\theta(\phi^{\ell}_{ij,t}), \qquad \varepsilon = 10^{-4}. \tag{2}$$

The rule MLP has input size 9 (six scalars and a three-entry layer indicator), two hidden layers of size 128 with ReLU nonlinearities, and three outputs.

The weight-update proposal updates the connection weight as

$$W^{(\ell)}_{ij,t+1} = W^{(\ell)}_{ij,t} - |\alpha| g^{W,\ell}_{ij,t}, \tag{3}$$

where the meta-learned scalar α, initialized at 10^{-3}, sets the learning rate shared by $W^{(1)}$, $W^{(2)}$ and the readout during meta-training.

The remaining two outputs of the rule MLP propose changes to the plasticity states ξ. For each neuron, proposals from its connections are averaged before being applied as

$$\xi'^{(\ell)}_{t,i} = \xi^{(\ell)}_{t,i} + \underbrace{\frac{1}{n_{\ell-1}} \sum_j g^{\mathrm{post},\ell}_{ij}}_{\text{updates from incoming connections}}, \qquad \xi^{(\ell)}_{t+1,i} = \xi'^{(\ell)}_{t,i} + \underbrace{\frac{1}{n_{\ell+1}} \sum_k g^{\mathrm{pre},\ell+1}_{ki}}_{\text{updates from outgoing connections}}. \tag{4}$$

Here, n_ℓ is the number of neurons in layer ℓ, and $\xi^{(\ell)}_{t,i}$ is the plasticity state of neuron i in layer ℓ at episode step t. Matrices $W^{(1)}$ and $W^{(2)}$ are updated in forward order, using activations from a single forward pass (Appendix B.1).

The readout weights are updated at every episode step by the supervised delta rule

$$W^{\mathrm{out}}_{t+1} = W^{\mathrm{out}}_t - |\alpha| \left(\mathrm{softmax}(\mathbf{o}_t) - \mathrm{onehot}(y_t)\right) (\mathbf{a}^{(2)}_t)^\top, \tag{5}$$

where $\mathbf{a}^{(2)}_t$ is the vector of activations in the final hidden layer. During online learning, the label-specific errors update only the readout and are not propagated to $W^{(1)}$ and $W^{(2)}$.

3.2 Meta-training

We independently meta-train ten rules with plasticity states and ten without. Each meta-training run contains 200,000 episodes, each of 512 episode steps (Algorithm 1, compute in Appendix B.2). Each episode draws a new pMNIST or RHM task with equal probability. Appendix B gives the meta-training hyperparameters.

Figure 2 shows the mean cross-entropy loss per episode over meta-training, with and without plasticity states.

3.3 Comparison and evaluation

We compare MLSP to online backpropagation, to readout-only learning and to the local Hebb, Oja, BCM and three-factor Hebb rules (Hebb, 1949; Oja, 1982; Bienenstock et al., 1982).

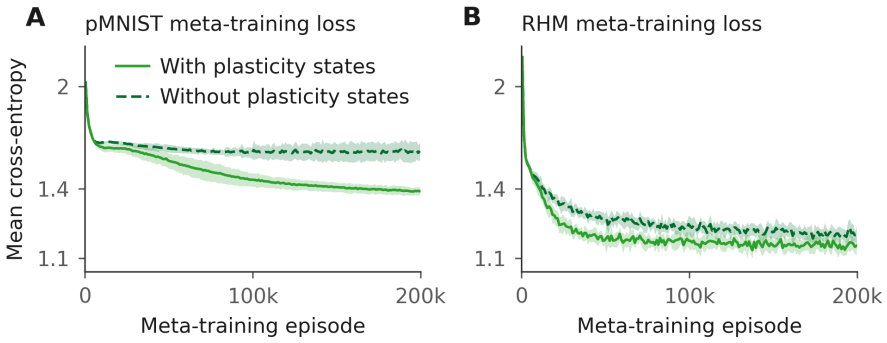

Figure 2: Meta-training loss on (A) pMNIST and (B) RHM. Losses are averaged separately by task family within bins of 1,000 meta-training episodes for each rule. Lines show the mean and bands show one standard deviation across ten rules with plasticity states and ten rules without.

Algorithm 1 MLSP meta-training.

The meta-parameters are the rule parameters θ, the shared rate α and the initial plasticity state ξ_{init}.
for each of 200,000 meta-training iterations **do**
  Initialize task-network weights P, $W^{(1)}$, $W^{(2)}$ and W^{out}, and set $\xi_{1,i}^{(\ell)} \leftarrow \xi_{\text{init}}$.
  Draw a 512-step pMNIST or RHM episode with equal probability.
  for $t = 1, \ldots, 512$ **do**
    Run the task-network forward pass and save $\mathbf{a}_t^{(0)}$, $\mathbf{a}_t^{(1)}$, $\mathbf{a}_t^{(2)}$ and $\mathbf{o}_t$.
    Observe y_t, compute the reward r_t (Equation 1) and record the cross-entropy of $\mathbf{o}_t$ against y_t.
    for $\ell = 1, 2$ in order **do**
      Apply the rule's weight and plasticity state updates for $W^{(\ell)}$, using the saved activations (Equations 3–4).
    end for
    Update W^{out} by Equation 5, using the saved $\mathbf{o}_t$ and $\mathbf{a}_t^{(2)}$.
  end for
  Differentiate the mean recorded cross-entropy through the episode and take one Adam step on θ, α and ξ_{init}.
end for

The classical hidden-layer rules are

$$\text{Hebb} \quad \Delta W_{ij}^{(\ell)} = \eta_\ell a_i^{(\ell)} a_j^{(\ell-1)}, \tag{6}$$

$$\text{three-factor Hebb} \quad \Delta W_{ij}^{(\ell)} = \eta_\ell (2r_t - 1) a_i^{(\ell)} a_j^{(\ell-1)}, \tag{7}$$

$$\text{Oja} \quad \Delta W_{ij}^{(\ell)} = \eta_\ell \big(a_i^{(\ell)} a_j^{(\ell-1)} - (a_i^{(\ell)})^2 W_{ij}^{(\ell)}\big), \tag{8}$$

$$\text{BCM threshold} \quad \theta_{i,t+1}^{(\ell)} = \theta_{i,t}^{(\ell)} + \frac{c(a_{i,t}^{(\ell)})^2 - \theta_{i,t}^{(\ell)}}{\tau_\theta}, \tag{9}$$

$$\text{BCM weights} \quad \Delta W_{ij,t}^{(\ell)} = \eta_\ell a_{j,t}^{(\ell-1)} a_{i,t}^{(\ell)} \big(a_{i,t}^{(\ell)} - \theta_{i,t+1}^{(\ell)}\big). \tag{10}$$

Here $W_{ij}^{(\ell)}$ connects neuron j in layer $\ell-1$ to neuron i in layer ℓ. Their current activations are $a_j^{(\ell-1)}$ and $a_i^{(\ell)}$. Each rule adds $\Delta W_{ij}^{(\ell)}$ to this weight, with learning rate η_ℓ. The reward r_t is 1 for a correct prediction at episode step t and 0 otherwise. For BCM, $\theta_{i,t}^{(\ell)}$ is the postsynaptic neuron's plasticity threshold before its update at episode step t (Appendix C.2). The threshold scale c and averaging timescale τ_θ are selected separately for each task family.

Although MLSP has no neuron-specific feedback, we include feedback alignment (Lillicrap et al., 2016) as a feedback-based rule that avoids the weight transport problem. While online backpropagation is not biologically plausible, it provides a comparison with access to true gradients throughout the network.

We select learning rates separately for each task family and rule. We search over separate learning rates for updates to $W^{(1)}$, $W^{(2)}$ and W^{out}. We select rates by mean accuracy at episode step 512, which we call step-512 accuracy (Appendix C.2).

4 RESULTS

4.1 PERFORMANCE AND GENERALIZATION ACROSS TASK FAMILIES

Figure 3A-D shows step-512 accuracy for four task families, including Fashion-MNIST and Omniglot, which are not used in meta-training. RHM meta-training samples a range of parameter settings. Figure 3B uses one setting, with a new hierarchy each episode (Appendix B). MLSP matches or outperforms every local rule on all four task families. At its learned rate rather than the selected rates, the MLSP rule performs as well on pMNIST, RHM and Omniglot, and only Fashion-MNIST benefits from rate selection (Appendix C.3). BCM is the best classical rule, but it is also the only one with extra parameters tuned to each task family, its threshold scale and timescale (Appendix C.2).

Figure 3E shows that rules with plasticity states gain more over readout-only learning than rules without them on all four task families. These gains use each rule's own (not searched) learning rates, so they are not directly comparable with A-D. On pMNIST, the absolute accuracies with and without plasticity states are $66.9 \pm 0.5\%$ and $57.8 \pm 1.3\%$, and Table 4 gives the other task families.

Figure 3F shows final permuted MNIST accuracy over a sweep of shared hidden-layer learning rates, with the readout learning rate optimized at each point. Hebb and Oja do not improve on readout-only learning at any tested hidden-layer learning rate. Across task families (Table 3), feedback alignment improves on readout-only learning only on RHM, and three-factor Hebb only on the image task families.

4.2 FITTING A POLYNOMIAL REPLACEMENT TO THE RULE WITHOUT PLASTICITY STATES

A rule parameterized by an MLP is hard to interpret. We test whether a polynomial can replace the meta-learned MLP rule. For simplicity, we study the rule without plasticity states, though we also attempt to fit the stateful rule in Appendix D.4. We fit a polynomial to a single meta-trained rule then learn with it in place of that rule. The polynomial has three variables, presynaptic activations a_{pre}, postsynaptic activations a_{post} and the current weight W, combined up to the second order. In this basis we get one constant, three 1st order terms, six 2nd order terms and one third order Oja-decay term. The reward signal is not included because a good fit is obtained without it and removing it during meta-training costs little (Figure 5B). Let ψ contain these 11 polynomial terms. The fitted replacement is

$$\begin{aligned} \widehat{g}_{ij}^{W,\ell} &= \sum_{k=1}^{11} c_{\ell,k} \psi_k(a_{\text{pre},j}, a_{\text{post},i}, W_{ij}^{(\ell)}), \\ \widehat{\Delta W}_{ij,t}^{(\ell)} &= -\eta_\ell \widehat{g}_{ij}^{W,\ell}. \end{aligned} \tag{11}$$

Here $c_{\ell,k}$ are fitted separately for each matrix. For Figure 4, the polynomial uses the source rule's learned or calibrated learning rates.

The coefficients are fitted with an L1 penalty to recorded rule inputs and weight-update proposals from 500 pMNIST and 500 RHM episodes. On 500 further episodes per task family, R^2 is 0.646 and 0.740 for the two matrices, and 0.475 for $W^{(1)}$ on RHM (Appendix D.1). Run in place of the rule, the polynomial retains 44–97% of the step-512 accuracy gain over readout-only learning of the rule without plasticity states (Figure 4C-F), and fits to all ten such rules have similar coefficients (Appendix D.3).

While a Hebbian term $a_{\text{pre}} a_{\text{post}}$ appears in the top five fitted terms for $W^{(1)}$, we do not observe clear structure of known classical rules such as Oja's rule, in contrast to some prior work (Confavreux et al., 2020). This is consistent with Hebb and Oja not improving on readout-only learning in this online setting of 512 episode steps (Figure 3). We study how including the Oja term affects the fit in Appendix D.2.

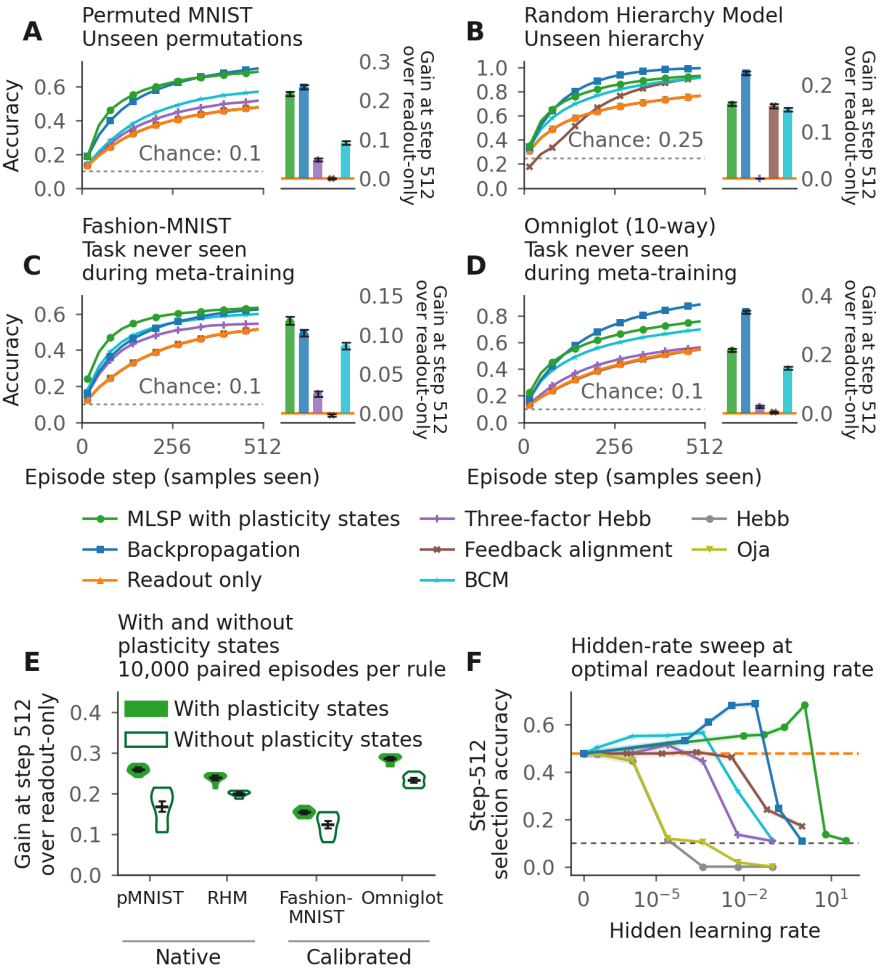

Figure 3: (A-D) Accuracy in 32-step bins across 10,000 evaluation episodes for each of the four task families. Bands show mean ± 1 standard error of the mean. Adjacent bars show the mean step-512 accuracy minus the corresponding readout-only result, with ± 1 SEM over episode differences. For A-D we use the MLSP rule from a single meta-training run. (E) Paired step-512 accuracy gain over a readout-only counterpart using 10,000 evaluation episodes per task family. Separately calculated for ten rules with plasticity states and ten rules without. Each counterpart keeps that condition's readout rate. Native uses learned rates and Calibrated uses learned rates times one multiplier per rule (Appendix C.4). The error bars show the SEM of the combined mean of the rules. (F) Learning rate sweep on pMNIST, showing step-512 selection accuracy while sweeping one common hidden rate and selecting the optimal readout rate. Bands show ± 1 SEM over 1,000 selection episodes.

4.3 Analysis of the plasticity states

For the first set of tests in Figure 5A, we intervene on a rule that has already been meta-trained. We record the plasticity states and reward signal during ordinary learning, then restart from the same initial network on the same samples while perturbing signals during learning. First we shuffle the time order of the recorded plasticity states. Next we shuffle the time order of the recorded reward signal. Third we reassign the plasticity states among neurons within each layer at each episode step.

Shuffling the plasticity state time ordering in Figure 5A reduces accuracy on both pMNIST and RHM substantially. Shuffling reward-signal time order has little effect, implying that the rule is effectively unsupervised. Reassigning the plasticity states among neurons within a layer at each episode step also has little effect.

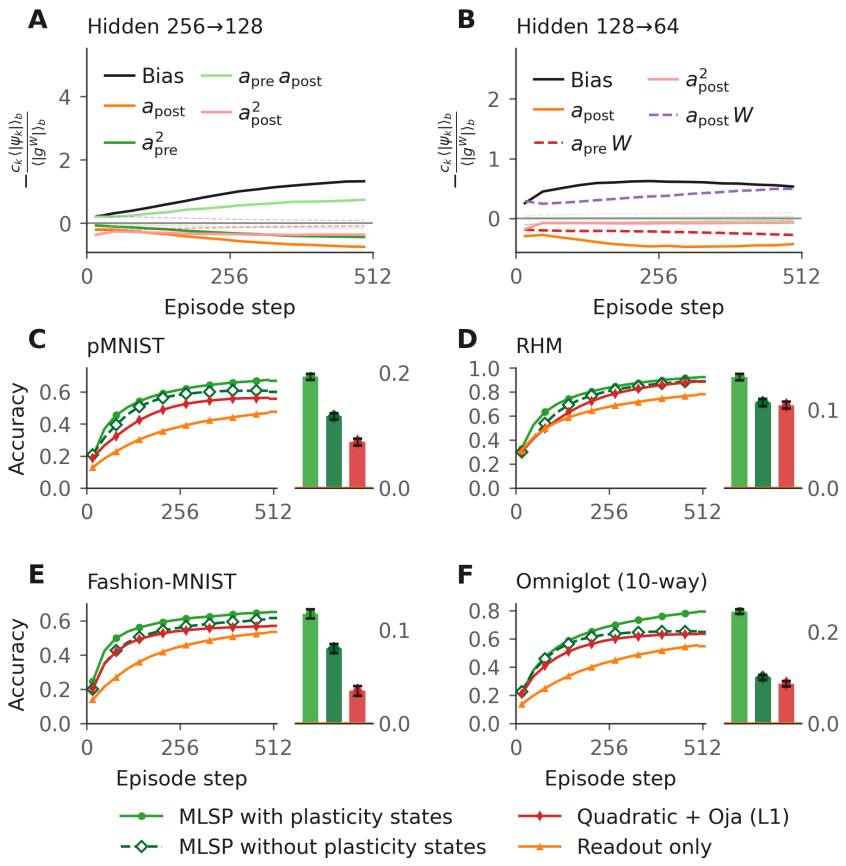

Figure 4: Polynomial up to the second order in weight, pre- and postsynaptic activations, plus the third order Oja-decay term $a^2_{\text{post}}W$, fitted to the rule inputs and weight-update proposals with an L1 penalty on the coefficients. (A,B) Term contributions over 100 pMNIST episodes, with sign $-\operatorname{sign}(c_k)$. Five largest mean absolute terms highlighted. The contribution of the term is normalized by the mean absolute weight-update proposal in each 32-step bin b (Appendix D.1). Bands show ± 1 SE over episodes, with the fit held fixed. Dashed curves contain an odd power of W and so follow the sign of W. Otherwise the sign reflects the contribution to the rule. (C-F) Accuracy in 32-step bins over 10,000 paired episodes per task family for the rule with plasticity states, the rule without plasticity states, its polynomial replacement and readout-only learning. Bands show ± 1 SEM.

For the second set of experiments in Figure 5B, we meta-train separate rules while withholding one input from the rule throughout meta-training (Appendix E.3). This tests which rule inputs matter most when learning a local rule. Removing postsynaptic activations produces the largest mean loss among the inputs tested. Removing the weight value causes the next largest drop in accuracy. Other ablations produce smaller mean accuracy losses.

We meta-trained five step-index controls without plasticity states but with an episode-step input. This episode-step input comes within one percentage point of the rules with plasticity states on pMNIST and RHM, three on Omniglot and nine on Fashion-MNIST (full results in Appendix E.2). This supports our hypothesis that the rules use the plasticity states mainly to represent progress through the episode.

5 Discussion and Limitations

MLSP finds rules that can support effective online learning and generalize. MLSP supports online learning on all four task families, including two absent from meta-training. The single rule eval-

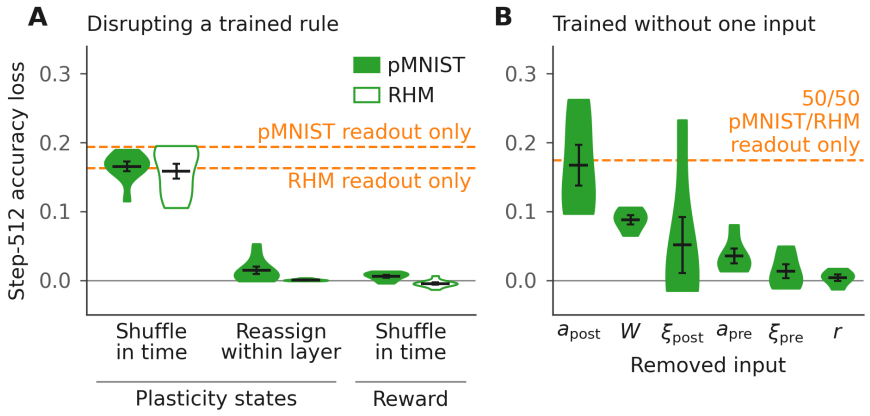

Figure 5: (A) Accuracy loss from interventions on ten meta-trained stateful rules, in which recorded plasticity states or rewards are shuffled in time, or plasticity states are reassigned among neurons within a layer, on paired evaluation episodes. (B) Accuracy loss for rules meta-trained with one input absent, on a 50/50 mixture of the pMNIST and RHM task families (five rules per input). ξ_{pre} and ξ_{post} are the pre- and postsynaptic plasticity states. Violin plots show the per-rule losses, and black lines and error bars show the mean across rules ± 1 SEM. Orange dashed lines give a reference, the intact accuracy minus the readout-only accuracy in Figure 3A,B, which B averages over both task families.

uated in Figure 3A–D approaches online backpropagation on pMNIST ($69.4 \pm 0.5\%$ vs $71.1 \pm 0.5\%$) and slightly exceeds it on Fashion-MNIST ($63.5 \pm 0.5\%$ vs $61.9 \pm 0.5\%$, Table 3), whereas classical rules such as Hebb and Oja do not improve over applying the delta rule to learn readout weights. Polynomial replacements of the MLSP-learned rules contain Hebbian and weight-dependent terms but do not reduce to a classical rule. In prior work Confavreux et al. (2020) recovered Oja-like rules when meta-learning for principal-component extraction, whereas our contribution shows that rapid online classification favors a different solution.

The hidden-layer updates depend little on reward. Our rules barely use reward. Shuffling it in meta-trained rules or removing it during meta-training has little effect, so during online learning the hidden layers learn effectively without supervision, like meta-learned unsupervised update rules (Metz et al., 2019). The label acts through the delta-rule readout instead. Future work should consider a reinforcement learning setting without a supervised delta rule, where the reward is required.

Plasticity states act as a layer-level learning schedule. Across ten meta-training runs per design, plasticity states improve accuracy. Shuffling them in time impairs learning, reassigning them within a layer has little effect, and an episode-step input recovers much of their benefit. Multiplying a fitted polynomial by a fitted exponential decay in time recovers much of the performance of a stateful rule (Appendix D.4). These results indicate that the rules use the plasticity states mainly to represent progress through the plasticity episode, a simple form of metaplasticity in which a neuron's history changes its subsequent plasticity (Abraham & Bear, 1996). This schedule is tied to the meta-training horizon of 512 episode steps. For the first-seed rule at its learned rate, pMNIST accuracy falls from $70.4 \pm 0.8\%$ at episode step 512 to $13.5 \pm 0.6\%$ at episode step 1,024, although resetting or clamping the plasticity states at episode step 512 largely prevents the decline (Appendix F).

Limitations and future work. We meta-trained on two task families with a fixed horizon of 512 episode steps. A broader set of meta-training task families will be needed to establish how general these solutions are. Applied without retraining at half and double hidden width, the stateful rules still learn on every task family, with the largest losses at half width on RHM and Omniglot (Appendix F.2). Rules could also be meta-learned with competition between the incoming synapses of each neuron. The delta-rule readout gives the output layer a label-specific error. Future work may meta-learn rules without it, in which case credit assignment would have to come from the reward alone. Future work should also extend MLSP to sign-constrained excitatory and inhibitory task networks.

AI USE STATEMENT

We used large language models to discuss research ideas and experimental designs, to help write, review and run code, and to interpret results. We also used them to find and check literature, draft and edit sections of the manuscript, and prepare figures and tables. We did not use generative AI to generate synthetic datasets or prove mathematical claims. The authors take responsibility for the final text, code, and results.

ETHICS STATEMENT

We use publicly released benchmark tasks. We collect no new participant data and do not deploy models to make decisions about people.

REPRODUCIBILITY STATEMENT

The supplementary ZIP contains anonymized training, evaluation and plotting code, trained checkpoints, training histories, polynomial fits and result summaries. The README gives setup instructions and commands to rerun the experiments. Large per-episode records are omitted to meet the 100 MB limit, so some figures require new evaluation or fitting runs to regenerate. A full GitHub repository will be made available after deanonymization.

REFERENCES

Wickliffe C. Abraham and Mark F. Bear. Metaplasticity: the plasticity of synaptic plasticity. *Trends in Neurosciences*, 1996. doi: 10.1016/s0166-2236(96)80018-x.

Mohamed Akrout, Collin Wilson, Peter C. Humphreys, Timothy Lillicrap, and Douglas B. Tweed. Deep learning without weight transport. In *Advances in Neural Information Processing Systems*, 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/hash/f387624df552cea2f369918c5e1e12bc-Abstract.html.

Marcin Andrychowicz, Misha Denil, Sergio Gómez Colmenarejo, Matthew W. Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando de Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems*, 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/hash/fb87582825f9d28a8d42c5e5e5e8b23d-Abstract.html.

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*, 2020. doi: 10.1038/s41467-020-17236-y.

Elie L. Bienenstock, Leon N Cooper, and Paul W. Munro. Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex. *The Journal of Neuroscience*, 1982. doi: 10.1523/jneurosci.02-01-00032.1982.

Francesco Cagnetta, Leonardo Petrini, Umberto M. Tomasini, Alessandro Favero, and Matthieu Wyart. How deep neural networks learn compositional data: the random hierarchy model. *Physical Review X*, 2024. doi: 10.1103/physrevx.14.031001.

Basile Confavreux, Friedemann Zenke, Everton J. Agnes, Timothy Lillicrap, and Tim P. Vogels. A meta-learning approach to (re)discover plasticity rules that carve a desired function into a neural network. In *Advances in Neural Information Processing Systems*, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/hash/bdbd5ebfde4934142c8a88e7a3796cd5-Abstract.html.

Basile Confavreux, Poornima Ramesh, Pedro J. Gonçalves, Jakob H. Macke, and Tim P. Vogels. Meta-learning families of plasticity rules in recurrent spiking networks using simulation-based inference. In *Advances in Neural Information Processing Systems*, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/2bdc2267c3d7d01523e2e17ac0a754f3-Abstract-Conference.html.

Nicolas Frémaux, Henning Sprekeler, and Wulfram Gerstner. Functional requirements for reward-modulated spike-timing-dependent plasticity. *The Journal of Neuroscience*, 2010. doi: 10.1523/jneurosci.6249-09.2010.

Ian J. Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. In *International Conference on Learning Representations*, 2014. URL https://arxiv.org/abs/1312.6211.

Stephen Grossberg. Competitive learning: from interactive activation to adaptive resonance. *Cognitive Science*, 1987. doi: 10.1111/j.1551-6708.1987.tb00862.x.

Jordan Guerguiev, Timothy P. Lillicrap, and Blake A. Richards. Towards deep learning with segregated dendrites. *eLife*, 2017. doi: 10.7554/elife.22901.

Donald O. Hebb. *The organization of behavior: a neuropsychological theory*. John Wiley & Sons, New York, 1949. URL https://pure.mpg.de/pubman/item/item_2346268.

Eugene M. Izhikevich. Solving the distal reward problem through linkage of STDP and dopamine signaling. *Cerebral Cortex*, 2007. doi: 10.1093/cercor/bhl152.

Diederik P. Kingma and Jimmy Lei Ba. Adam: a method for stochastic optimization. In *International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6980.

Louis Kirsch and Jürgen Schmidhuber. Meta learning backpropagation and improving it. In *Advances in Neural Information Processing Systems*, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/7608de7a475c0c878f60960d72a92654-Abstract.html.

Daniel Kunin, Aran Nayebi, Javier Sagastuy-Brena, Surya Ganguli, Jonathan M. Bloom, and Daniel L. K. Yamins. Two routes to scalable credit assignment without weight symmetry. In *Proceedings of the 37th International Conference on Machine Learning*, 2020. URL https://proceedings.mlr.press/v119/kunin20a.html.

Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 2015. doi: 10.1126/science.aab3050.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998. doi: 10.1109/5.726791.

Dong-Hyun Lee, Saizheng Zhang, Asja Fischer, and Yoshua Bengio. Difference target propagation. In *Machine Learning and Knowledge Discovery in Databases*, 2015. doi: 10.1007/978-3-319-23528-8_31.

Timothy P. Lillicrap, Daniel Cownden, Douglas B. Tweed, and Colin J. Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 2016. doi: 10.1038/ncomms13276.

Jack Lindsey and Ashok Litwin-Kumar. Learning to learn with feedback and local plasticity. In *Advances in Neural Information Processing Systems*, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/hash/f291e10ec3263bd7724556d62e70e25d-Abstract.html.

Niru Maheswaranathan, David Sussillo, Luke Metz, Ruoxi Sun, and Jascha Sohl-Dickstein. Reverse engineering learned optimizers reveals known and novel mechanisms. In *Advances in Neural Information Processing Systems*, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/a57ecd54d4df7d999bd9c5e3b973ec75-Abstract.html.

Luke Metz, Niru Maheswaranathan, Brian Cheung, and Jascha Sohl-Dickstein. Meta-learning update rules for unsupervised representation learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HkNDsiC9KQ.

Thomas Miconi, Jeff Clune, and Kenneth O. Stanley. Differentiable plasticity: training plastic neural networks with backpropagation. In *Proceedings of the 35th International Conference on Machine Learning*, 2018. URL https://proceedings.mlr.press/v80/miconi18a.html.

Elias Najarro and Sebastian Risi. Meta-learning through Hebbian plasticity in random networks. In *Advances in Neural Information Processing Systems*, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/hash/ee23e7ad9b473ad072d57aaa9b2a5222-Abstract.html.

Arild Nøkland. Direct feedback alignment provides learning in deep neural networks. In *Advances in Neural Information Processing Systems*, 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/hash/d490d7b4576290fa60eb31b5fc917ad1-Abstract.html.

Erkki Oja. Simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology*, 1982. doi: 10.1007/bf00275687.

Alexandre Payeur, Jordan Guerguiev, Friedemann Zenke, Blake A. Richards, and Richard Naud. Burst-dependent synaptic plasticity can coordinate learning in hierarchical circuits. *Nature Neuroscience*, 2021. doi: 10.1038/s41593-021-00857-x.

João Sacramento, Rui Ponte Costa, Yoshua Bengio, and Walter Senn. Dendritic cortical microcircuits approximate the backpropagation algorithm. In *Advances in Neural Information Processing Systems*, 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/hash/1dc3a89d0d440ba31729b0ba74b93a33-Abstract.html.

Mark Sandler, Max Vladymyrov, Andrey Zhmoginov, Nolan Miller, Andrew Jackson, Tom Madams, and Blaise Agüera y Arcas. Meta-learning bidirectional update rules. In *Proceedings of the 38th International Conference on Machine Learning*, 2021. URL https://proceedings.mlr.press/v139/sandler21a.html.

Megha Sehgal, Chenghui Song, Vanessa L. Ehlers, and James R. Moyer, Jr. Learning to learn – intrinsic plasticity as a metaplasticity mechanism for memory formation. *Neurobiology of Learning and Memory*, 2013. doi: 10.1016/j.nlm.2013.07.008.

Navid Shervani-Tabar and Robert Rosenbaum. Meta-learning biologically plausible plasticity rules with random feedback pathways. *Nature Communications*, 2023. doi: 10.1038/s41467-023-37562-1.

Danil Tyulmankov, Guangyu Robert Yang, and L. F. Abbott. Meta-learning synaptic plasticity and memory addressing for continual familiarity detection. *Neuron*, 2022. doi: 10.1016/j.neuron.2021.11.009.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747, 2017. URL https://arxiv.org/abs/1708.07747.

A TERMINOLOGY

We use the following terminology to distinguish the task network from the plasticity rule, and online learning from meta-training.

Model components.

Meta-Learned Stateful Plasticity (MLSP). A framework for meta-training a shared local plasticity rule and using the fixed rule to train newly initialized task networks online. MLSP refers to the framework, not to either neural network alone.

Task network. The feedforward classifier that learns a classification task within one plasticity episode. In our experiments, it consists of the nonplastic input projection P, the hidden weight matrices $W^{(1)}$ and $W^{(2)}$, and the readout weights W^{out}. The task network is newly initialized at the beginning of each episode. Its three hidden layers, $\ell = 0, 1, 2$, are connected by the hidden weight matrices $W^{(1)}$ and $W^{(2)}$.

Nonplastic input projection. The random projection P from the input to the task network's first hidden layer. It remains fixed throughout a plasticity episode but is redrawn at the beginning of the next episode. We use *nonplastic input projection*, rather than *fixed input weights*, because it is fixed only within an episode.

Hidden weight matrices. The task-network matrices $W^{(1)}$ and $W^{(2)}$ that are updated online by the local plasticity rule. A scalar element $W_{ij}^{(\ell)}$ is a *connection weight*.

Readout weights. The output matrix W^{out}, which maps the activations of the final hidden layer of the task network to the logits. It is updated at each episode step by a supervised delta rule using the revealed label. Its error signal is not propagated to the hidden weight matrices under MLSP.

Plasticity rule. The shared function g_θ that maps the information available at one plastic connection to proposals for updating that connection and the plasticity states of its endpoint neurons. Its parameters remain fixed during a plasticity episode.

Rule MLP. The multilayer perceptron used to parameterize the plasticity rule g_θ. The same rule MLP is applied independently at every connection in the hidden weight matrices. It is not the task network and does not itself perform classification.

Rule parameters. The weights and biases θ of the rule MLP. They are optimized during meta-training and remain fixed during both online learning and evaluation. We use *rule parameters*, rather than *MLP weights*, to avoid confusion with the task-network weights.

Meta-parameters. The quantities optimized during meta-training. These are the rule parameters θ, the shared learning rate α, and, for stateful rules, the initial plasticity-state value ξ_{init}.

Plasticity state. A writable scalar $\xi_{t,i}^{(\ell)}$ carried by one hidden neuron of the task network. It can be read and updated by the plasticity rule and can retain information between episode steps. It does not enter the feedforward computation and is reset to ξ_{init} at the beginning of each episode.

Rule inputs. The quantities supplied to the rule MLP at one plastic connection. They are the presynaptic activation, postsynaptic activation, current connection weight, plasticity states of the two connected neurons, broadcast binary reward signal and layer indicator.

Reward. The binary scalar r_t that equals one when the task network's prediction for the current sample is correct and zero otherwise. It is computed from the prediction made before the current update and is broadcast identically to all plastic connections.

Update proposal. One of the three outputs produced by the rule MLP at a plastic connection, namely a weight-update proposal and two plasticity-state-update proposals, one for each endpoint neuron. The weight-update proposal is scaled by $|\alpha|$ before it is applied. Plasticity-state-update proposals are averaged separately over a neuron's incoming and outgoing connections, and both averages are added to its plasticity state without scaling by $|\alpha|$.

Neuron-specific feedback. A teaching or error signal whose value differs across hidden neurons and is derived from the output error, as in backpropagation or feedback alignment. MLSP

does not provide neuron-specific feedback to the hidden weight matrices during online learning, although the readout receives supervised label information.

Learning processes.

Task family. A distribution over related classification problems. We use pMNIST, RHM, Fashion-MNIST and Omniglot.

Task instance. One particular classification problem sampled from a task family, such as one pixel permutation in pMNIST or one sampled hierarchy in RHM. The dataset-specific definition of a task instance is stated separately for each task family.

Plasticity episode. One sequence of online learning on a single task instance. At the beginning of an episode, the task network is newly initialized, the nonplastic input projection is redrawn, and the plasticity states are reset. The episode then presents 512 samples sequentially. The hidden weight matrices, readout weights and plasticity states change during the episode, whereas the rule parameters and other meta-parameters remain fixed. This is often called the *inner loop* in meta-learning.

Episode step. One position t in a plasticity episode. At each episode step, the task network makes a prediction, the label is revealed, the reward is computed, and the hidden weight matrices, plasticity states and readout weights are updated. Accuracy at episode step 512, the last step of an episode, is written step-512 accuracy. The classification samples do not form a dynamical input sequence.

Online learning. Learning from samples presented one at a time, with an update after every sample and no mini-batches or training epochs. In MLSP, online learning happens within a plasticity episode, in which $W^{(1)}$, $W^{(2)}$, W^{out} and ξ change but the rule parameters θ do not.

Meta-training loss. The mean cross-entropy over all 512 episode steps of a plasticity episode. Each episode step's loss is computed from the prediction made before using that sample's label to update the task network.

Meta-training run. One complete and independently initialized sequence of plasticity episodes used for meta-training. Each run has 200,000 plasticity episodes and produces one independently meta-trained MLSP rule. Different meta-training runs of one design are not different architectures.

Meta-trained rule. The plasticity rule obtained at the end of one meta-training run. Once meta-training is complete, its rule parameters, α and ξ_{init} are frozen for evaluation.

Evaluation episode. A plasticity episode used to measure a fixed learning rule, with no meta-parameter update.

Variants, evaluation, and analysis.

Rules with and without plasticity states. A stateful rule, or rule with plasticity states, reads and updates the plasticity states of its endpoint neurons. A rule without plasticity states has no plasticity-state inputs or outputs and updates only the hidden weight matrices.

Step-index control. A rule without plasticity states that receives the normalized episode step, $(t-1)/511$, as an additional input. This control tests whether plasticity states improve performance primarily by representing progress through the episode.

Transfer. Evaluation on a task family that did not contribute to meta-training, such as Fashion-MNIST or Omniglot.

Paired evaluation episodes. Evaluation episodes in which two methods or conditions receive the same task instance, sample order and task-network initialization. Pairing isolates differences caused by the learning rules or interventions.

Readout-only learning. A baseline in which only W^{out} is updated during the episode, while $W^{(1)}$ and $W^{(2)}$ keep their initial values.

Online backpropagation. A baseline in which exact gradients of the current supervised loss update $W^{(1)}$, $W^{(2)}$ and W^{out} at each episode step. This baseline has access to neuron-specific error information that is unavailable to MLSP.

Evaluation-time intervention. A perturbation applied to the inputs or states of an already meta-trained rule during a paired evaluation episode. The rule parameters are kept fixed. Examples include shuffling the plasticity states or reward in time and reassigning plasticity states among neurons within a layer.

Input removal during meta-training. A new meta-training run in which one rule input is set to zero throughout meta-training. This tests whether meta-learning can discover an alternative rule without that input. It is distinct from an evaluation-time intervention on a previously meta-trained rule.

Polynomial replacement. A polynomial function fitted to the weight-update proposals of a meta-trained rule and then used in place of the rule MLP during evaluation. The replacement in Section 4.2 is fitted with an L1 penalty to the rule without plasticity states. It contains all monomials up to second order in presynaptic activations, postsynaptic activations and the current connection weight, plus the Oja-decay monomial $a_{\mathrm{post}}^2 W$, giving 11 terms per matrix (Appendix D.1).

B Implementation and meta-training

B.1 Initialization and update order

Each episode initializes P, $W^{(1)}$, $W^{(2)}$ and W^{out} with PyTorch's default Kaiming-uniform scheme. All 448 plasticity states (256, 128 and 64 neurons in layers 0, 1 and 2) start at ξ_{init}. The rule MLP has biases in all three linear layers. Each rule call reads the pre-update weights of its matrix. The call for $W^{(1)}$ writes to layers 0 and 1. The call for $W^{(2)}$ reads the updated layer-1 plasticity states, then writes to layers 1 and 2.

B.2 Meta-training protocol

Each MLSP design has ten meta-training runs (seeds 42–51), giving rules 1–10. Each run has 200,000 episodes of 512 episode steps. Single-rule analyses use rule 1, from the first seed. Tasks, weights and plasticity states reset each episode. The proposal scale $\varepsilon = 10^{-4}$ was chosen empirically to keep early meta-training stable and is the same in every run. Adam (Kingma & Ba, 2015) uses learning rate 10^{-4}, default momentum parameters and no weight decay. Gradients are norm-clipped at 1.0. Forward passes use bfloat16 autocast; losses accumulate in float32.

The objective is mean pre-update cross-entropy over the episode (Algorithm 1). Gradients do not cross episode boundaries or pass through the binary reward's argmax. The rule MLP has 18,179 parameters with plasticity states and 17,665 without. The final $|\alpha|$ ranges from 1.020 to 1.181 across the ten rules with plasticity states (first seed: 1.165) and from 1.024 to 1.681 across the ten without (first seed: 1.225).

Compute. Each meta-training run used one GPU on a shared cluster. Jobs were limited to three days, so runs resumed from checkpoints and could move between GPU types, mostly NVIDIA A100, L40S, RTX A4500 and Quadro RTX 5000. A run of 200,000 episodes took 157–185 GPU-hours (6.5–7.7 days) when it ran only on A100s. There are 55 meta-training runs reported here (ten with and ten without plasticity states, five step-index controls and five for each of six removed inputs).

B.3 Meta-training task families

Episodes draw pMNIST or RHM with equal probability. MNIST training images are sampled with replacement and normalized with mean 0.1307 and SD 0.3081. Each episode uses one pixel permutation and one affine transform. Rotation, translation and scale are uniform on $[-15^\circ, 15^\circ]$, $[-2, 2]$ pixels per axis, and $[0.9, 1.1]$. Gaussian noise is drawn per pixel and sample. Its SD is drawn uniformly from $[0, 0.2]$ once per episode.

RHM labels depend on a compositional hierarchy of features, which tests whether the rule can use hierarchical structure with local plasticity in the hidden layers. RHM draws $v \in \{4, 8, 16\}$, $L \in \{2, 3\}$, $n_c \in \{2, 3, 4, 5, 6, 8, 10\}$, and $m \in \{2, 3, 4\}$, with $s = 2$. Here v counts feature types,

L is hierarchy depth, and n_c counts classes. Each feature has m possible constructions from s sub-features. Draws must fit the network dimensions and satisfy $v^s \geq m \max(v, n_c)$. Some parameter combinations are therefore rejected. Inputs are one-hot encoded, flattened and zero-padded to 784 values, without image normalization. We use the generator's training partition. It withholds 20% of possible patterns, rounded down and capped at 20,000. Sampling uses replacement only when fewer than 512 patterns remain.

B.4 Evaluation episodes and metrics

Table 1 describes how evaluation episodes are sampled. Every RHM evaluation uses a fresh hierarchy with the configuration in Table 1, and Omniglot always uses its evaluation split. Table 2 lists the rules, rates, image split and episode count of each result. All task families use 16 logits, so expected initial accuracy is $1/16$. Dotted chance lines mark $1/K$, uniform guessing among the K valid labels. Evaluation uses no image augmentation and never changes the rule parameters.

Table 1: Evaluation task families.

Task family	Episode sampling	Relation to meta-training
pMNIST	MNIST images; 10 classes; samples with replacement; one pixel permutation per episode	pMNIST task family used in meta-training
RHM	Fresh hierarchy per episode with $v = 16, L = 2, n_c = 4, m = 3, s = 2$; generator training partition (87 of 108 patterns); samples with replacement	Configuration within the meta-training range
Fashion-MNIST	10 classes; samples with replacement; one pixel permutation per episode; MNIST-normalized	No meta-gradient; monitored during development
Omniglot	Evaluation split (659 characters in 20 alphabets); 10 characters chosen at random per episode, relabeled 0–9; samples drawn with replacement from their pooled images; images resized to 28×28, inverted, and MNIST-normalized	No meta-training episodes; not inspected during development

Because P has i.i.d. entries and is redrawn each episode, a pixel permutation does not change the distribution of the projected input. New permutations alone therefore do not test distributional transfer. Samples are drawn with replacement from a pool of N items, so the input at episode step 512 has appeared earlier in the episode with probability $1 - (1 - 1/N)^{511}$: 0.997 on RHM ($N = 87$), 0.923 on Omniglot ($N = 200$) and at most 0.050 on MNIST and Fashion-MNIST ($N \geq 10{,}000$). On RHM and Omniglot, step-512 accuracy therefore mostly scores items already seen in the episode.

Figure 3A–D and Figure 4 measure gains against readout-only learning at a selected or calibrated readout rate. Figure 3E and Appendix D.4 instead use the rule's own readout rate, which differs from the readout-only optimum, so their gains are not comparable with those of Figure 3A–D. We compare the designs using the absolute accuracies in Table 4.

Performance is step-512 accuracy, or step-1,024 accuracy for the longer episodes of Appendix F. Trajectories use nonoverlapping 32-step bins within each episode. Endpoints remain unbinned. After a numerical failure (non-finite activations, logits, weights or plasticity states), the remaining predictions of that episode count as incorrect. No episode is excluded.

Error bars show ± 1 SEM unless stated otherwise; Figure 2 shows SD across trained rules. Paired differences are computed within episodes first. For comparisons across trained rules, we resample rules and shared episodes. The SD of these bootstrap means gives the SEM. For polynomial fits, we resample whole episodes without refitting the polynomial. The SD of the resulting fit statistics gives their bootstrap SE.

Table 2: Rules, learning rates, image split and episodes for each result. Rules are meta-trained with plasticity states unless stated, and 10+10 means ten rules with and ten without plasticity states. Split applies to pMNIST and Fashion-MNIST. Episodes are per task family and rule. *The rules with plasticity states use 3,000 on Fashion-MNIST and Omniglot.

Result	Rules	Learning rates	Split	Episodes
Figure 3A–D, Table 3	First seed	Selected per task family and method (Appendix C.2)	Test	10,000
Figure 3E, Table 4	10+10	Learned; one multiplier per rule on Fashion-MNIST and Omniglot (Appendix C.4). Readout-only uses the rule's readout rate	Training	10,000
Figure 3F	First seed	Equal hidden rates from the searches in A (Appendix C.2)	Test	1,000
Figure 10	First seed	Learned (1.165) vs selected	Test	10,000
Figure 4, Table 6, Appendix D.2	First seed with and first seed without plasticity states	Learned on pMNIST, calibrated on the other task families; readout-only calibrated (Table 5)	Training	10,000
Figure 13C–F, Table 7	First seed	Learned (1.165), also for readout-only	Training	10,000
Figure 5A	10	Learned	Training	10,000
Figure 5B, Table 8	First five seeds for each removed input	Learned	Training	10,000
Figure 16	10, plus 5 step-index controls	Learned	Training	10,000*
Figure 17	First seed	Learned	Test	3,000
Ten-rule horizon screen (Appendix F)	10	Learned	Test	100
Figure 18	10	Learned	Training	3,000

C PERFORMANCE COMPARISONS

C.1 FEEDBACK ALIGNMENT AND BACKPROPAGATION

All comparison methods train the readout with Equation 5 at their selected readout rate. Feedback alignment uses fixed random feedback matrices drawn each episode. With $e_t = \mathrm{softmax}(\mathbf{o}_t) - \mathrm{onehot}(y_t)$,

$$\delta_2 = (B_{\mathrm{out}}^\top e_t) \odot \mathbf{1}[\mathbf{a}_t^{(2)} > 0],$$
$$\delta_1 = (B_2^\top \delta_2) \odot \mathbf{1}[\mathbf{a}_t^{(1)} > 0],$$
$$\Delta W^{(\ell)} = -\eta_\ell \delta_\ell (\mathbf{a}_t^{(\ell-1)})^\top.$$

Here $B_{\mathrm{out}} \in \mathbb{R}^{16\times 64}$ and $B_2 \in \mathbb{R}^{64\times 128}$ are independent Kaiming-uniform draws; $\odot$ denotes elementwise multiplication and the indicators give the ReLU derivatives. Backpropagation computes all three matrix gradients from the same forward pass before changing any weights. The input projection remains fixed for both methods.

C.2 RATE SELECTION FOR FIGURE 3A–D AND F

Panels A–D use one MLSP rule (the first seed) and seven comparison methods. For each task family and method, including MLSP, we select the rates that maximize mean step-512 accuracy over 1,000 selection episodes. The selected rates are evaluated on 10,000 separate reporting episodes. All methods share samples and initial weights within each episode.

The search varies η_1, η_2 and η_{out} independently. Each hidden-rate grid has zero and six log-spaced positive values: 10^{-4} to 1 for backpropagation, 10^{-6} to 1 for feedback alignment, 10^{-7} to 10^{-1} for Hebb, Oja, three-factor Hebb and BCM, and 10^{-2} to $10^{1.5}$ for MLSP. The readout grid has zero and seven log-spaced values, from 10^{-3} to $10^{0.5}$ on image task families and from 10^{-2} to 10^{2} on RHM. BCM also searches $\tau_\theta \in \{1, 10, 10^2, 10^3, 10^4\}$ and $c \in \{0.25, 0.5, 1, 2, 4\}$. A positive boundary optimum triggers a tenfold extension of that axis. Extensions are skipped when the lowest positive rate ties with a zero rate or when selection accuracy is 1. No reported selection has an unresolved improving boundary.

The BCM threshold is initialized to $c(a_{i,1}^{(\ell)})^2$ on the first episode step, and the weight update uses the updated threshold $\theta_{i,t+1}^{(\ell)}$. With $\tau_\theta = 1$, the BCM threshold equals $c(a_{i,t}^{(\ell)})^2$ at every episode step rather than a running average. The classical rules update each neuron's incoming weights without competition or normalization across neurons, and three-factor Hebb uses the reward term $2r_t - 1$ without a reward baseline.

Table 3 lists the rates and reporting accuracies. For BCM, (τ_θ, c) is $(10, 1)$ on pMNIST, $(1, 40)$ on RHM, $(1, 1)$ on Fashion-MNIST and $(1, 2)$ on Omniglot. For BCM, we therefore tune both the rule and its rates for each task family. On Omniglot, Hebb and Oja select $\eta_1 = 10^{-7}$ by a margin of one correct prediction in 1,000 selection episodes; their reporting accuracy, 0.551, is within one SEM of readout-only learning (0.552).

Figures 6–9 show the completed selection grids. Each cell gives the highest step-512 accuracy over the tested readout rates and, for BCM, threshold parameters at that hidden-rate pair. These are selection scores; Table 3 reports performance on separate episodes.

Panel F uses the equal-hidden-rate cells, $\eta_1 = \eta_2$, of the pMNIST searches above. At each tested hidden rate, we maximize step-512 accuracy over the readout rate and, for BCM, the threshold parameters. Each point uses 1,000 selection episodes; bands show ± 1 SEM. MLSP uses the same rule as panels A–D.

C.3 MLSP AT ITS LEARNED RATE

Figure 10 repeats Figure 3A–D for MLSP alone, at its selected rates and at its learned rate, $|\alpha| = 1.165$ for both hidden weight matrices and the readout, on the same 10,000 reporting episodes. Selection changes step-512 accuracy by $+0.000 \pm 0.002$ on pMNIST, -0.002 ± 0.001 on RHM, $+0.066 \pm 0.005$ on Fashion-MNIST and -0.006 ± 0.005 on Omniglot (selected minus learned,

Table 3: Selected rates and step-512 accuracies for Figure 3A–D. Rate triples are $(\eta_1, \eta_2, \eta_{\mathrm{out}})$, rounded to three significant figures. Accuracies are means ± 1 SEM over 10,000 reporting episodes. MLSP uses one trained rule.

Task family	Method	Selected rates	Accuracy
pMNIST	MLSP (first seed)	$(1.26, 1.26, 0.825)$	0.694 ± 0.005
	Backpropagation	$(0.0251, 6.31\times10^{-4}, 0.00383)$	0.711 ± 0.005
	Readout only	$(0, 0, 0.0562)$	0.477 ± 0.005
	Three-factor Hebb	$(2.51\times10^{-5}, 2.51\times10^{-5}, 0.0562)$	0.526 ± 0.005
	BCM	$(3.98\times10^{-4}, 2.51\times10^{-5}, 0.0562)$	0.570 ± 0.005
	Feedback alignment	$(2.51\times10^{-4}, 2.51\times10^{-4}, 0.0562)$	0.478 ± 0.005
	Hebb	$(0, 0, 0.0562)$	0.477 ± 0.005
	Oja	$(0, 0, 0.0562)$	0.477 ± 0.005
RHM	MLSP (first seed)	$(1.26, 1.26, 1)$	0.929 ± 0.003
	Backpropagation	$(0.00398, 0, 21.5)$	0.994 ± 0.001
	Readout only	$(0, 0, 4.64)$	0.769 ± 0.004
	Three-factor Hebb	$(0, 0, 4.64)$	0.769 ± 0.004
	BCM	$(0.00631, 0.1, 100)$	0.917 ± 0.003
	Feedback alignment	$(1, 1.58\times10^{-5}, 0.0464)$	0.924 ± 0.003
	Hebb	$(0, 0, 4.64)$	0.769 ± 0.004
	Oja	$(0, 0, 4.64)$	0.769 ± 0.004
Fashion-MNIST	MLSP (first seed)	$(0.251, 0.251, 0.215)$	0.635 ± 0.005
	Backpropagation	$(0.00398, 0.00398, 1\times10^{-5})$	0.619 ± 0.005
	Readout only	$(0, 0, 0.0147)$	0.516 ± 0.005
	Three-factor Hebb	$(2.51\times10^{-5}, 2.51\times10^{-5}, 0.0562)$	0.541 ± 0.005
	BCM	$(0, 2.51\times10^{-5}, 0.0562)$	0.603 ± 0.005
	Feedback alignment	$(2.51\times10^{-4}, 1\times10^{-6}, 0.0147)$	0.514 ± 0.005
	Hebb	$(0, 0, 0.0147)$	0.516 ± 0.005
	Oja	$(0, 0, 0.0147)$	0.516 ± 0.005
Omniglot	MLSP (first seed)	$(0.01, 0.251, 0.825)$	0.767 ± 0.004
	Backpropagation	$(0.0251, 0, 0.00383)$	0.897 ± 0.003
	Readout only	$(0, 0, 0.0562)$	0.552 ± 0.005
	Three-factor Hebb	$(2.51\times10^{-5}, 2.51\times10^{-5}, 0.215)$	0.574 ± 0.005
	BCM	$(2.51\times10^{-5}, 3.98\times10^{-4}, 0.825)$	0.706 ± 0.005
	Feedback alignment	$(2.51\times10^{-4}, 0.00398, 0.0562)$	0.554 ± 0.005
	Hebb	$(1\times10^{-7}, 0, 0.0562)$	0.551 ± 0.005
	Oja	$(1\times10^{-7}, 0, 0.0562)$	0.551 ± 0.005

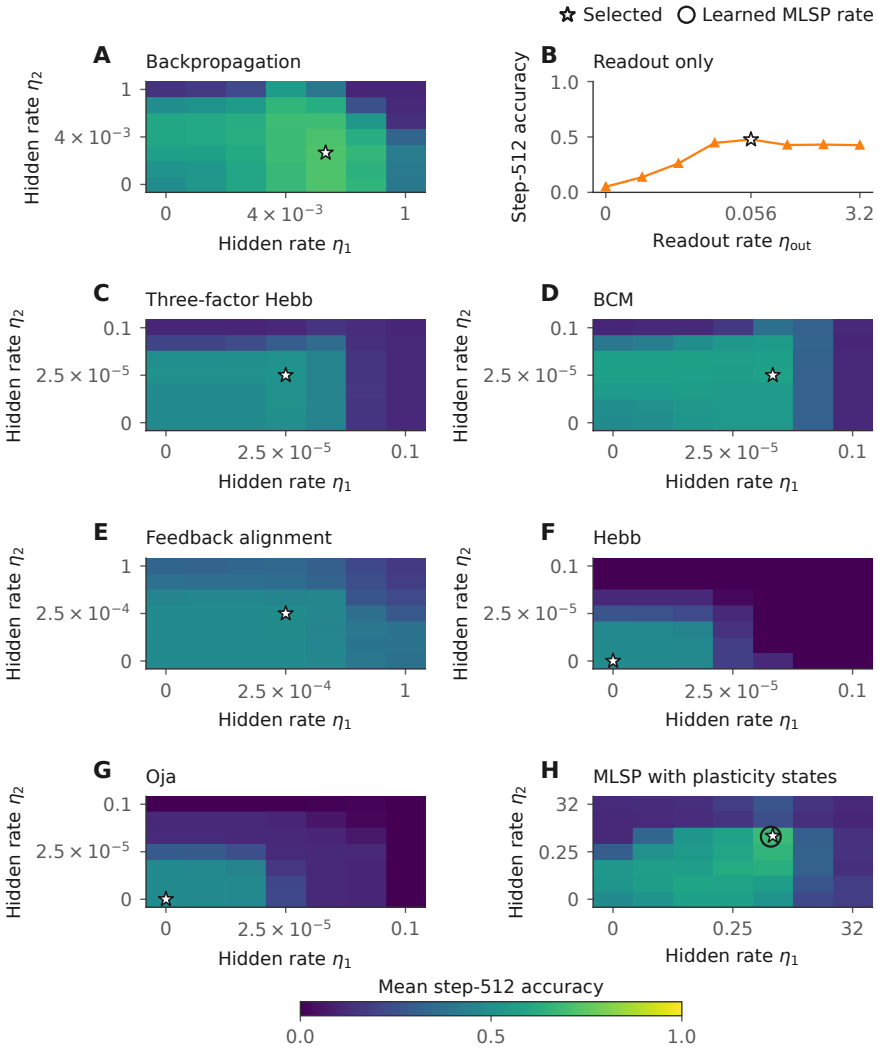

Figure 6: pMNIST rate selection for Figure 3A: mean step-512 accuracy over 1,000 selection episodes. Heatmaps maximize over the readout rate and, for BCM, the threshold parameters. Stars mark selected settings; the circle marks the learned MLSP rate. Readout-only error bars show ± 1 SEM.

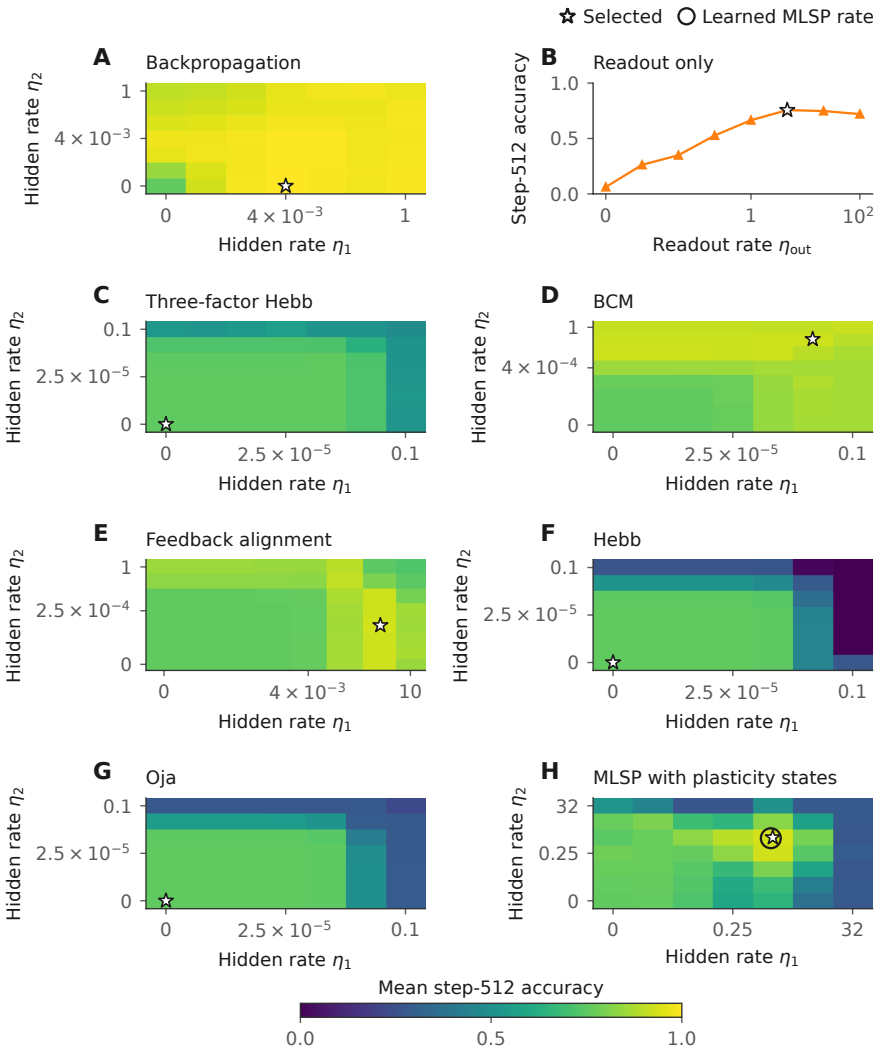

Figure 7: RHM rate selection for Figure 3B. The metric, episode count and markers follow Figure 6. Search ranges include boundary extensions.

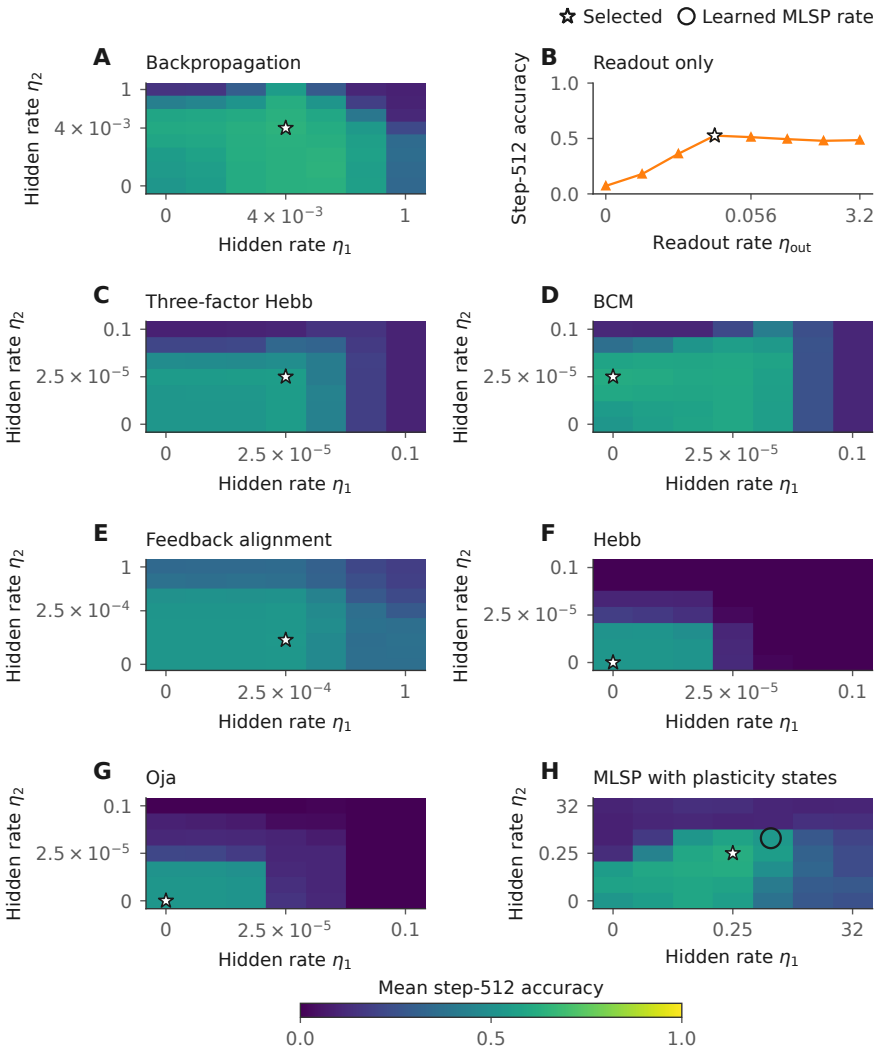

Figure 8: Fashion-MNIST rate selection for Figure 3C. The metric, episode count and markers follow Figure 6. Search ranges include boundary extensions.

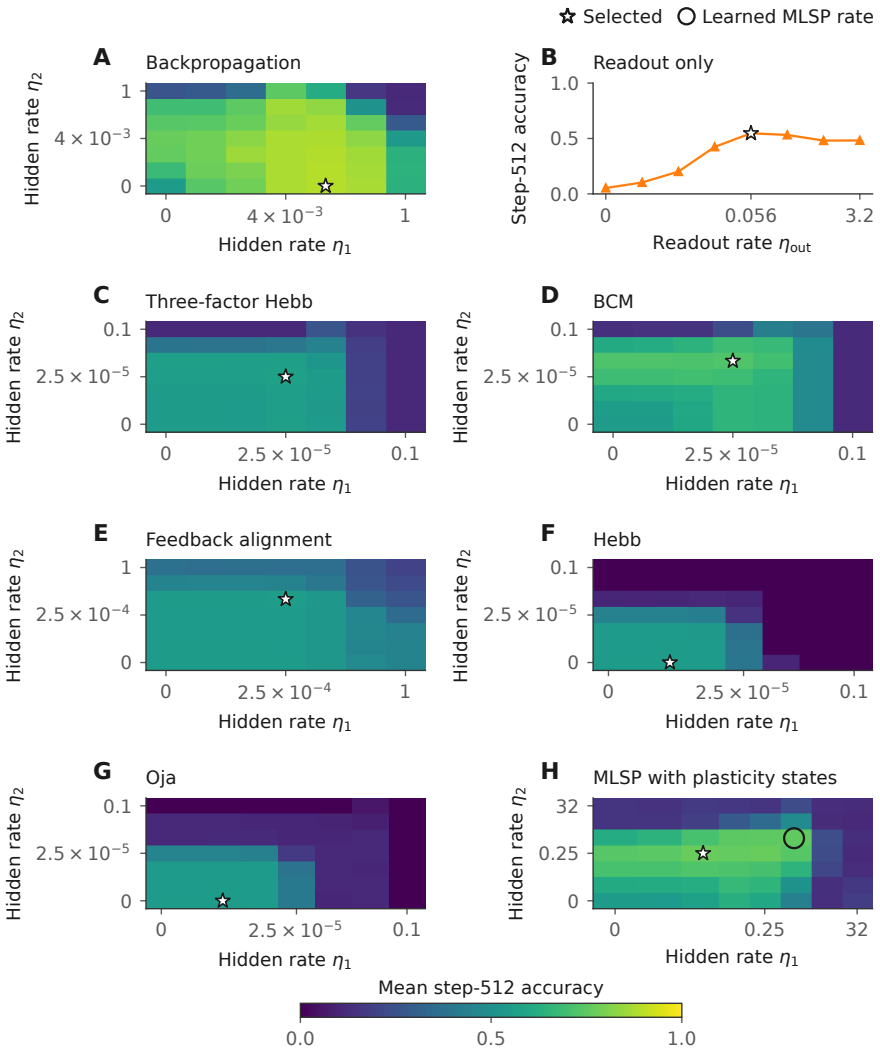

Figure 9: Omniglot rate selection for Figure 3D. The metric, episode count and markers follow Figure 6. Search ranges include boundary extensions.

paired, ± 1 SEM). The learned rate is therefore as good as the selected rates on pMNIST, RHM and Omniglot; only Fashion-MNIST gains from rate selection.

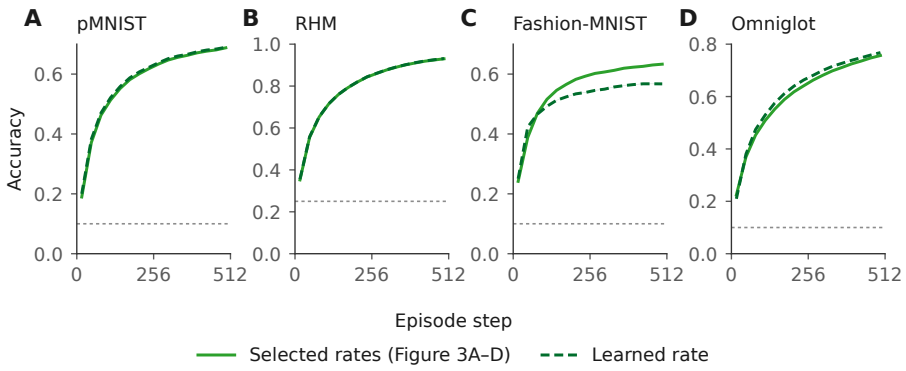

Figure 10: MLSP (first seed) at its selected rates (Figure 3A–D) and at its learned rate, on the same 10,000 test-split reporting episodes per task family. Accuracy in 32-step bins; bands show ± 1 SEM across episodes. Dotted lines show chance accuracy.

C.4 Rules with and without plasticity states (Figure 3E)

Figure 3E compares ten rules per design. Each rule's gain is measured against a readout-only counterpart with the same readout rate, on 10,000 paired episodes per task family. pMNIST and RHM use learned rates. On Fashion-MNIST and Omniglot, one multiplier scales a rule's learned rate for all three matrices. It is chosen from $\{0.06, 0.125, 0.25, 0.5, 0.71, 1, 1.41, 2, 4\}$ by final-50 accuracy (mean accuracy over the last 50 episode steps) on 12 separate episodes. The selected multipliers are 0.25–0.5 with plasticity states and 0.125–0.25 without on Fashion-MNIST, and 0.71–1 and 0.5–0.71 on Omniglot.

Table 4 gives the absolute accuracies. Rules with plasticity states have higher mean accuracy on all four task families.

Table 4: Step-512 accuracy for Figure 3E. Each design has ten meta-trained rules, with 10,000 evaluation episodes per task family and rule. Entries are means ± 1 bootstrap SEM over rules and shared episode indices; the two designs' rules are resampled independently. Difference: with minus without plasticity states.

Task family	With plasticity states	Without plasticity states	Difference
pMNIST	0.669 ± 0.005	0.578 ± 0.013	$+0.092$
RHM	0.925 ± 0.004	0.894 ± 0.003	$+0.031$
Fashion-MNIST	0.633 ± 0.005	0.605 ± 0.008	$+0.028$
Omniglot	0.770 ± 0.005	0.720 ± 0.006	$+0.049$

D Polynomial replacements

All fits use recorded rule inputs and weight-update proposals from 500 pMNIST and 500 RHM fitting episodes. We sample 12 synapses per matrix at each episode step, giving 6,144,000 recorded synapse updates per matrix. Separate episodes, 100 per task family, select the penalty (quadratic-plus-Oja fits) or give held-out fit statistics (cubic fits), as specified below. The fitted proposals include ε and the coefficients are used unscaled: hidden updates are $-\eta_\ell \widehat{g}_\ell^W$. Online tests use the first seed of each design, with fitting, validation and reporting on separate episodes.

D.1 QUADRATIC-PLUS-OJA FIT (FIGURE 4)

Figure 4 uses the reward-free basis in Equation 11: all monomials of total degree at most two in $(a_{\mathrm{pre}}, a_{\mathrm{post}}, W)$, plus the Oja-decay monomial $a_{\mathrm{post}}^2 W$. There are 11 coefficients per matrix, 22 in total. For each matrix we minimize

$$\frac{1}{2N}\sum_{n=1}^{N}\left(g_{\ell,n}^{W} - \mathbf{c}_\ell^\top \boldsymbol{\psi}_n\right)^2 + \lambda_\ell \sum_{k=2}^{11} |c_{\ell,k}|. \tag{12}$$

Here N is the number of recorded synapse updates, $\boldsymbol{\psi}_n$ contains the basis terms, and $g_{\ell,n}^{W}$ is the source rule's weight-update proposal. The L1 penalty is included to favor a sparse rule with few terms. It acts on raw-basis coefficients, and the constant coefficient $c_{\ell,1}$ is unpenalized. We select each matrix's positive penalty by pooled MSE on 100 pMNIST and 100 RHM episodes, obtaining $\lambda_1 = 4.56 \times 10^{-8}$ and $\lambda_2 = 1.70 \times 10^{-7}$. These 200 episodes select the penalty; they are not an independent fit test. Their mean squared proposal errors are 1.19×10^{-7} and 2.16×10^{-7} for the two matrices. At these penalties all 20 non-constant coefficients remain nonzero, so the fitted rule is not sparse in this basis. Stronger penalties remove terms only at a cost in fit, and the best fits with 7 of the 10 non-constant terms raise the selection MSE by 23% for $W^{(1)}$ and 6% for $W^{(2)}$. On 500 further pMNIST and 500 further RHM episodes, separate from fitting and penalty selection, the fit has R^2 of 0.646 and 0.740 for the two matrices (pMNIST: 0.720 and 0.755; RHM: 0.475 and 0.724).

Figure 4A,B uses the 100 pMNIST penalty-selection episodes. For each 32-step bin b, the plotted contribution of term k is

$$-\operatorname{sign}(c_{\ell,k})\frac{\langle |c_{\ell,k}\psi_k| \rangle_b}{\langle |g_\ell^W| \rangle_b}. \tag{13}$$

Angle brackets $\langle\cdot\rangle_b$ average over sampled synapses, episode steps in bin b and episodes. The minus sign follows the weight update. Terms with odd powers of W have dashed curves because their update direction also depends on W. Bands are bootstrap SE from resampling episodes with the coefficients fixed.

Rates. Fitting episodes use the source rule's learned rate, 1.225. Online, the polynomial uses exactly the rates of its source rule. On pMNIST, both MLSP designs and the polynomial use learned rates. On the other task families, each design uses layerwise rates calibrated by minimum summed pre-update cross-entropy (image task families) or maximum final-50 accuracy (RHM). Readout-only learning uses a separately calibrated readout rate on every task family. Calibration episodes are separate from reporting episodes and use the test split for pMNIST and Fashion-MNIST. Table 5 lists the rates; because they and the image split differ from those in Table 3, accuracies also differ. Table 6 gives the step-512 accuracies of Figure 4C–F.

Table 5: Rates $(\eta_1, \eta_2, \eta_{\mathrm{out}})$ in Figure 4, rounded. The polynomial uses the rates of MLSP without plasticity states.

Task family	MLSP with plasticity states	MLSP without plasticity states and polynomial	Readout only (η_{out})
pMNIST	$(1.165, 1.165, 1.165)$	$(1.225, 1.225, 1.225)$	0.0412
RHM	$(0.629, 0.903, 1.79)$	$(0.662, 0.950, 1.88)$	7.50
Fashion-MNIST	$(0.537, 0.386, 0.383)$	$(0.128, 0.0101, 0.110)$	0.0222
Omniglot	$(0.732, 0.866, 1.09)$	$(1.48, 1.25, 2.02)$	0.0767

D.2 REMOVING THE OJA TERM

To test the Oja-decay term $a_{\mathrm{post}}^2 W$, we refit the L1 basis without it for all ten rules without plasticity states, with the same fitting and penalty-selection episodes. On 500 further pMNIST and 500 further RHM episodes per rule, removing the term changes R^2 from 0.676 ± 0.011 to 0.652 ± 0.014 for $W^{(1)}$ and from 0.696 ± 0.016 to 0.637 ± 0.016 for $W^{(2)}$ (means ± 1 SEM over rules). For the first seed on the paired episodes of Figure 4, step-512 accuracy changes from 0.557 to 0.540 on pMNIST, 0.886 to 0.887 on RHM, 0.569 to 0.560 on Fashion-MNIST and 0.636 to 0.618 on Omniglot (SEM 0.005 or less). Including the term therefore has a small effect on the image task families and none on

Table 6: Step-512 accuracy for Figure 4: 10,000 paired episodes per task family, with one trained rule per design. Entries are means ± 1 SEM.

Task family	MLSP with plasticity states	MLSP without plasticity states	Polynomial	Readout only
pMNIST	0.670 ± 0.005	0.601 ± 0.005	0.557 ± 0.005	0.477 ± 0.005
RHM	0.922 ± 0.003	0.890 ± 0.003	0.886 ± 0.003	0.781 ± 0.004
Fashion-MNIST	0.651 ± 0.005	0.614 ± 0.005	0.569 ± 0.005	0.534 ± 0.005
Omniglot	0.793 ± 0.004	0.649 ± 0.005	0.636 ± 0.005	0.549 ± 0.005

RHM. Because it adds a parameter to each fit, a higher R^2 alone does not show that the rule contains an Oja-like term.

D.3 Coefficient consistency across rules

We repeat the fit of Appendix D.1 for all ten rules without plasticity states. Figure 11 shows the coefficients. Mean pairwise cosine similarities between the coefficient vectors are 0.874 and 0.972 for the two matrices (minimum 0.458 and 0.900). After weighting each coefficient by the mean absolute size of its term, $c_{\ell,k}\langle|\psi_k|\rangle$, they are 0.912 and 0.907 (minimum 0.706 and 0.689). The least similar fits are rule 8 for $W^{(1)}$ and rule 4 for weighted $W^{(2)}$ (Figure 12). Term sizes are averaged over the pMNIST and RHM fitting episodes of the first seed, whereas Figure 4A,B uses pMNIST only. Neither coefficient plot normalizes the vectors. These similarities compare fitted rule shapes, not online accuracy.

D.4 Stateful rules

These fits approximate the first seed of MLSP with plasticity states. All conditions, including readout-only learning, use its learned rate, 1.165, for both hidden weight matrices and the readout. Neither replacement uses reward or plasticity states updated during the episode. A constant cubic, with all 20 monomials of total degree at most three in $(a_{\mathrm{pre}}, a_{\mathrm{post}}, W)$, has held-out R^2 of 0.243 and 0.415 for the two matrices.

Modulated cubic. We instead multiply the constant cubic by a fitted time function for each matrix:

$$\widehat{g}_\ell(t) = f_\ell(t)\,\mathbf{c}_\ell^\top \boldsymbol{\psi}, \qquad f_\ell(t) = b_\ell + (1 - b_\ell)\exp\bigl(-k_\ell(t-1)/511\bigr), \quad t = 1, \ldots, 512. \tag{14}$$

Each matrix has 20 coefficients and two time-function parameters, with $f_\ell(1) = 1$ so that the coefficients set the initial scale. The search uses $0 \le b_\ell \le 1$ and $0 \le k_\ell \le 64$; for each candidate schedule, we fit the coefficients by least squares. The fitted plateau and decay time $(b_\ell, 511/k_\ell)$ are $(0.0082, 45.1)$ and $(0.0311, 15.5)$. Held-out R^2 rises to 0.708 and 0.796 (Figure 13B). The same coefficients and schedules are used on every task family. The modulated cubic preserves online learning (Figure 13C–F); it matches the source rule on Omniglot. It has numerical failures in 166 of 10,000 Fashion-MNIST episodes; the source rule, constant cubic and readout-only learning have none. Figure 14 shows its term contributions.

Table 7: Step-512 accuracy of replacements for the first seed of MLSP with plasticity states, over 10,000 paired episodes per task family. All conditions use the learned rate. Entries are means ± 1 SEM.

Task family	Source rule	Constant cubic	Modulated cubic	Readout only
pMNIST	0.669 ± 0.005	0.494 ± 0.005	0.572 ± 0.005	0.408 ± 0.005
RHM	0.930 ± 0.003	0.839 ± 0.004	0.910 ± 0.003	0.695 ± 0.005
Fashion-MNIST	0.580 ± 0.005	0.411 ± 0.005	0.553 ± 0.005	0.481 ± 0.005
Omniglot	0.771 ± 0.004	0.614 ± 0.005	0.771 ± 0.004	0.483 ± 0.005

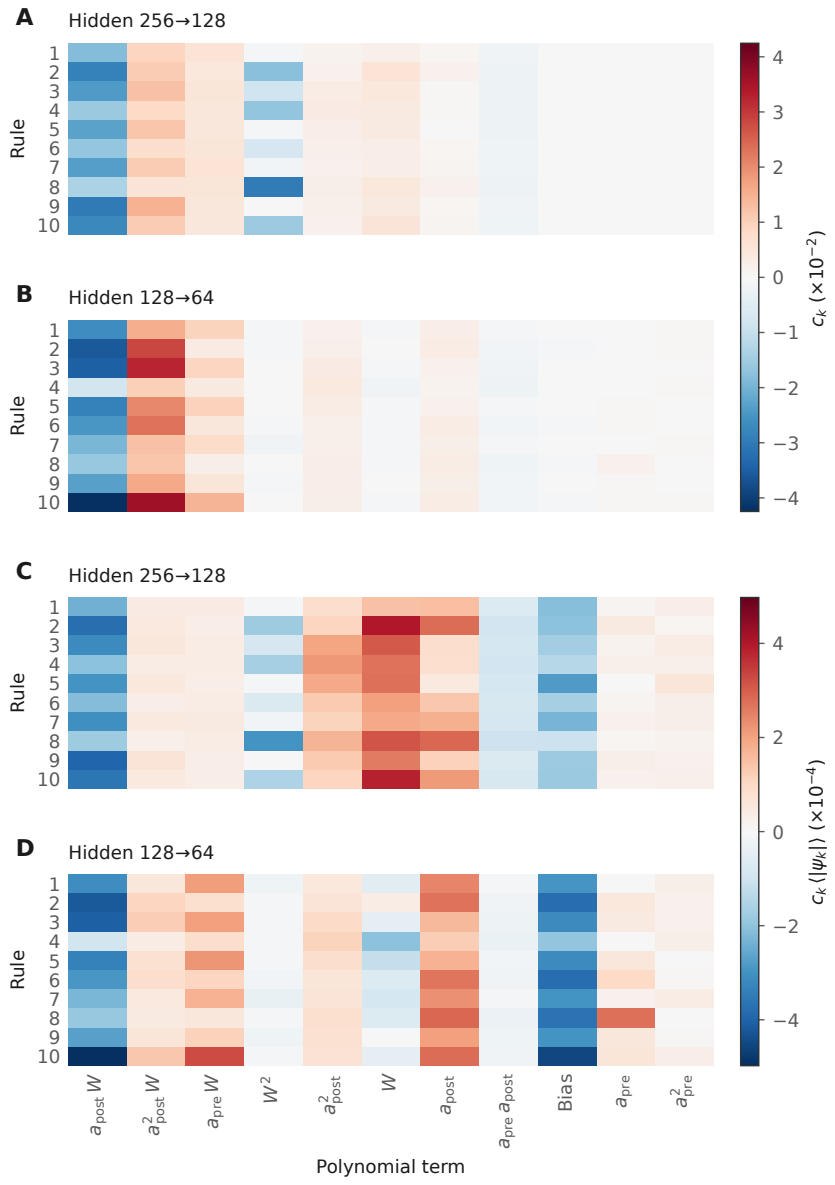

Figure 11: Quadratic-plus-Oja coefficients of ten rules without plasticity states. (A,B) Raw coefficients c_k. (C,D) Coefficients multiplied by mean absolute term size, $c_k\langle|\psi_k|\rangle$. Rows are rules and columns are polynomial terms. The only third-order term is $a^2_{\text{post}}W$.

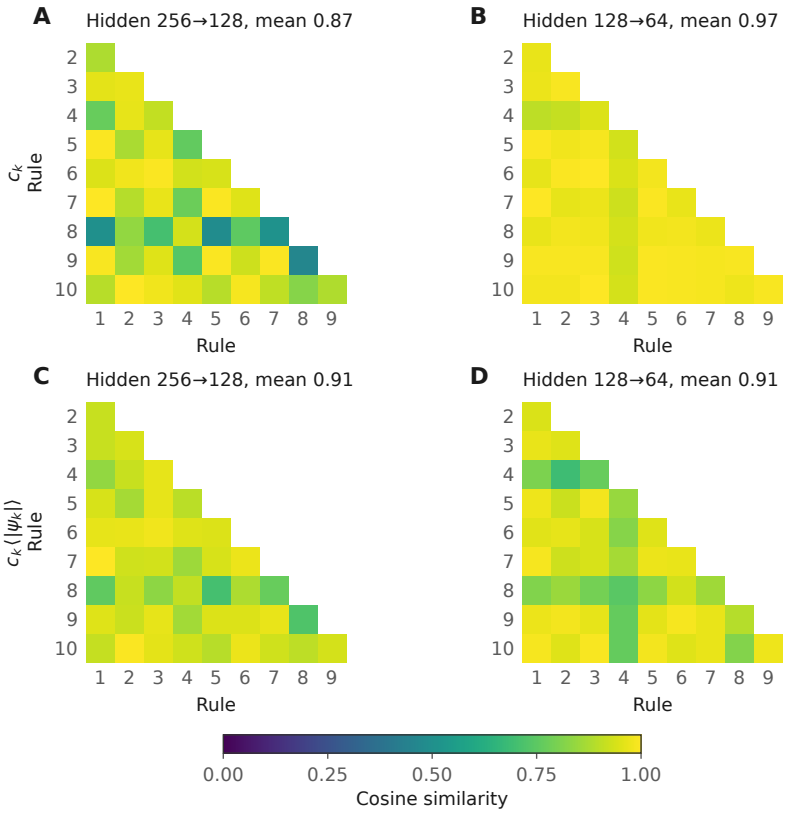

Figure 12: Cosine similarity between the coefficient vectors of Figure 11, for each pair of rules. (A,B) Raw coefficients. (C,D) Coefficients weighted by mean absolute term size. Panel titles give the mean over the 45 pairs.

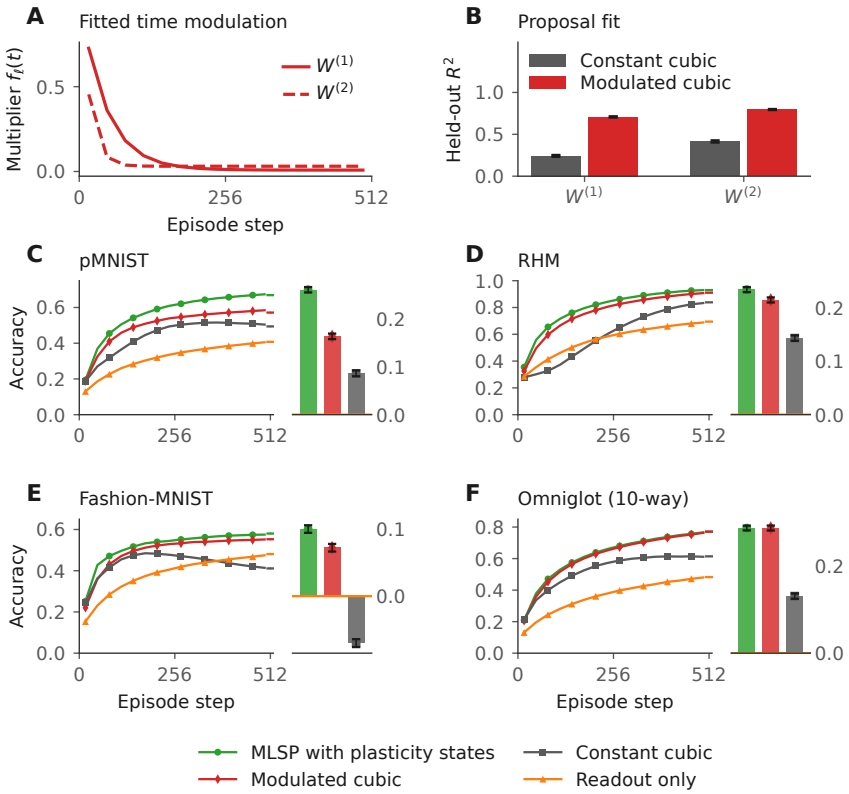

Figure 13: Modulated cubic. (A) Fitted schedules $f_\ell(t)$, as 32-step bin means. (B) Held-out proposal R^2 on 100 pMNIST and 100 RHM episodes, with ± 1 bootstrap SE from resampling episodes within each task family; the fits stay fixed. (C–F) Accuracy in 32-step bins over 10,000 paired episodes per task family. Bands show ± 1 SEM. Ticks at episode step 512 show unbinned accuracy ± 1 SEM. Adjacent bars show paired step-512 gains over readout-only learning, ± 1 SEM.

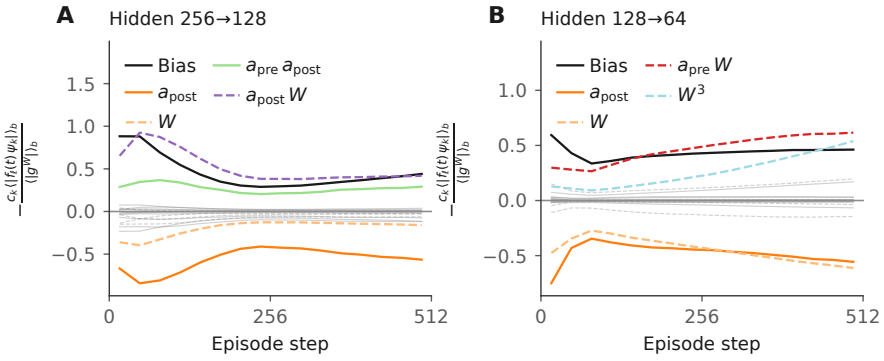

Figure 14: Term contributions to the modulated cubic. Each contribution includes $f_\ell(t)$ and is normalized as in Figure 4A,B; dashed curves contain an odd power of W. Averages use 32-step bins from 100 pMNIST and 100 RHM held-out episodes. The five largest mean absolute contributions are highlighted for each matrix; other terms are gray. Bands show ± 1 bootstrap SE over episodes, resampled within each task family with the fit fixed.

E PLASTICITY STATES AND RULE INPUTS

E.1 SHUFFLING AND REASSIGNING PLASTICITY STATES

For the first seed, the layer-mean plasticity states decline steadily over the episode (Figure 15). Figure 5A uses ten rules and 10,000 paired episodes per task family and rule. Each rule receives equal weight.

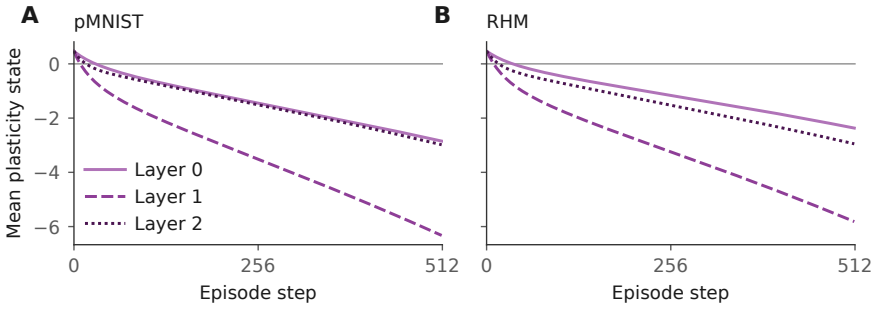

Figure 15: Plasticity states during learning. Layer-mean plasticity states of the first-seed stateful rule on (A) pMNIST and (B) RHM, read at the start of each episode step; layer 1 is read before its first write in each episode step. All three layers start at the learned initial value $\xi_{\text{init}} = 0.472$ and decline over the episode, with the largest decline in layer 1. Lines show means over 600 episodes per task family; ± 1 SEM bands are narrower than the lines.

Time-shuffle tests record the plasticity states and reward read at each rule call: layer 0, layer 1 before and after its first write, and layer 2. Reruns use identical samples and initial networks. Replay without shuffling reproduces the correctness of every prediction in the original episode. One time permutation per episode shuffles all four recorded plasticity state inputs. Reward shuffling changes only the reward order.

Reassignment instead permutes the current plasticity states before every episode step after the first. Values are reassigned among the neurons of each layer. Plasticity state updates continue normally. All neurons start with the same plasticity state, so reassignment before the first episode step would have no effect. Accuracy losses from within-layer reassignment are 0.014 ± 0.005 on pMNIST and 0.001 ± 0.001 on RHM.

E.2 STEP-INDEX CONTROL

We meta-train five rules without plasticity states (the first five seeds) that also receive the normalized episode step $(t-1)/511$ at both rule calls. The step index resets each episode. Their rule MLP has 17,793 parameters.

Figure 16 compares the step-index rules with the rules with plasticity states at learned rates. Mean step-512 accuracies with plasticity states and with the step index are 0.669 ± 0.005 and 0.660 ± 0.005 on pMNIST, 0.925 ± 0.004 and 0.923 ± 0.003 on RHM, 0.497 ± 0.017 and 0.403 ± 0.039 on Fashion-MNIST, and 0.766 ± 0.007 and 0.735 ± 0.005 on Omniglot. On Fashion-MNIST one step-index rule reaches only 0.241, and the other four average 0.443. Restricted to the same five seeds, the rules with plasticity states reach 0.670, 0.923, 0.518 and 0.765. All rules here use their learned rates, so Fashion-MNIST and Omniglot accuracies are lower than in Table 4, which uses calibrated rates on those task families.

E.3 INPUTS REMOVED DURING META-TRAINING

Each removed input has five separately trained rules. Figure 5B and Table 8 compare them with rules with all inputs from the same meta-training seeds (the first five). Each comparison uses 10,000 paired episodes per task family and rule. Task families and rules receive equal weight.

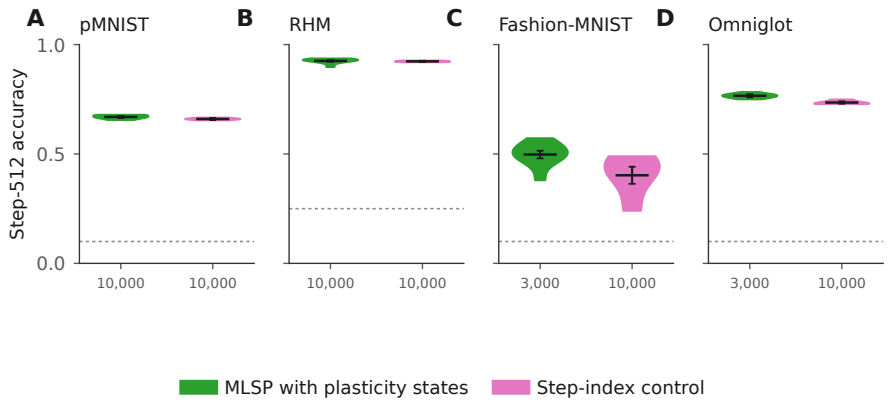

Figure 16: Step-index control. Ten MLSP rules with plasticity states and five step-index rules at learned rates. Each step-index rule uses 10,000 evaluation episodes per task family, as do the rules with plasticity states on pMNIST and RHM, which use 3,000 on Fashion-MNIST and Omniglot. Episodes are paired by index. Violins show the rule means; black lines and error bars show overall means ± 1 bootstrap SEM over rules and shared episode indices. Dotted lines show chance accuracy.

Meta-training can compensate for the missing input. Removing one plasticity state input leaves the other input and both plasticity-state-update outputs in place.

Table 8: Step-512 accuracy loss after meta-training without one input, relative to the rule with all inputs and the same meta-training seed. Each condition has five rules and 10,000 paired episodes per task family. Entries are means ± 1 bootstrap SEM over rules and paired episode indices.

Removed input	pMNIST	RHM	Equal mixture
Postsynaptic activations	0.241 ± 0.062	0.093 ± 0.033	0.167 ± 0.029
Weight	0.074 ± 0.006	0.102 ± 0.014	0.088 ± 0.007
Postsynaptic plasticity state	0.092 ± 0.079	0.011 ± 0.011	0.051 ± 0.041
Presynaptic activations	0.059 ± 0.009	0.012 ± 0.015	0.035 ± 0.011
Presynaptic plasticity state	0.029 ± 0.018	-0.002 ± 0.009	0.013 ± 0.010
Binary reward	0.003 ± 0.005	0.005 ± 0.006	0.004 ± 0.005

For the postsynaptic plasticity state, the largest rule-level equal-mixture loss is 0.233, while the other four average 0.006.

F Beyond the meta-training conditions

F.1 Longer episodes

We run the first seed of MLSP with plasticity states for 1,024 episode steps on 3,000 paired pMNIST test episodes and 3,000 paired RHM episodes at its learned rate (Figure 17). After the step-512 update, the episode continues intact, resets the plasticity states once to ξ_{init}, or clamps them at their step-512 values. Clamping also blocks writes between the two rule calls. Weights keep learning. All three conditions start from the same step-512 network and use the same remaining samples.

Accuracy is 0.704 at episode step 512. At episode step 1,024 it is 0.135 intact, 0.652 after reset and 0.701 after clamp. Paired gains over the intact condition are 0.517 ± 0.010 and 0.566 ± 0.010 (means ± 1 SEM).

On RHM, accuracy is 0.947 at episode step 512. The intact condition keeps improving until about episode step 750 and then declines to 0.637 at episode step 1,024, whereas reset and clamp give 0.919 and 0.951. Paired gains over the intact condition are 0.281 ± 0.010 and 0.314 ± 0.009.

A separate pMNIST comparison uses the first 100 paired episodes for each of ten rules. Every rule has a positive estimated gain from both interventions. Mean gains are 0.437 ± 0.073 after reset

and 0.396 ± 0.076 after clamp, with bootstrap SEM over rules and episodes. No numerical failures occur. These tests do not establish reliable learning beyond 1,024 episode steps.

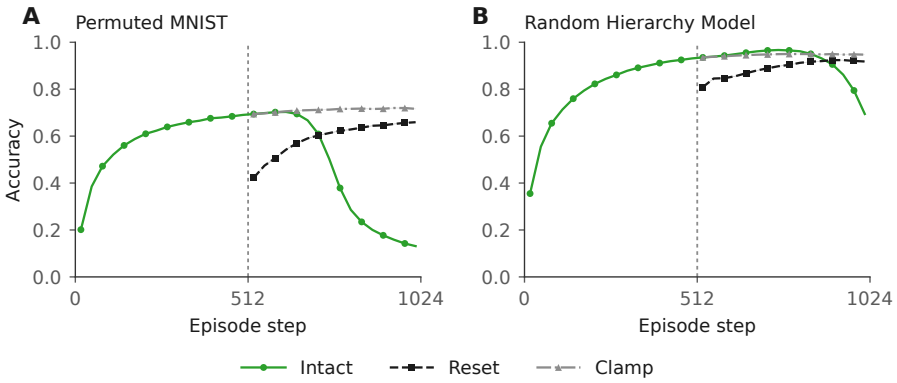

Figure 17: Learning beyond episode step 512. One rule on 3,000 paired episodes of (A) pMNIST and (B) RHM, in 32-step bins with ± 1 SEM across episodes. After the step-512 update (dotted line), the plasticity states are left intact, reset once, or clamped. Weights are not reset and rates stay fixed.

F.2 Other network widths

We apply the ten stateful rules at half, trained and double width: $128 \to 64 \to 32$, $256 \to 128 \to 64$ and $512 \to 256 \to 128$. Depth, readout size, rule parameters and learned rates stay fixed. Each condition uses 3,000 shared episodes per task family and rule. Width effects differ across task families (Figure 18). The largest estimated losses are at half width on RHM and Omniglot.

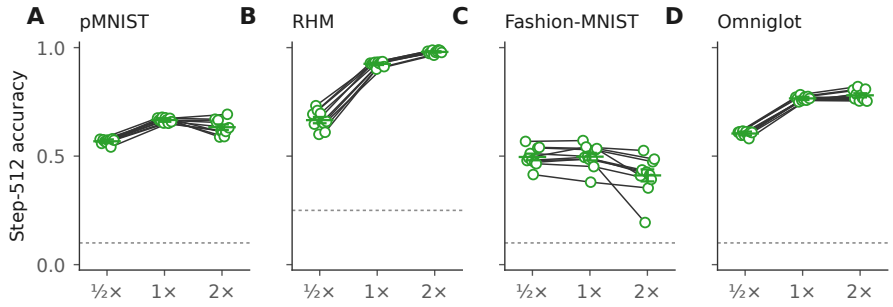

Figure 18: Other network widths for ten stateful rules. Lines join each rule's step-512 accuracies over 3,000 shared episodes per width and task family; $\frac{1}{2}\times$, $1\times$ and $2\times$ are half, trained and double width for the task-network hidden layers. Horizontal marks show means ± 1 bootstrap SEM over rules and episode indices. Rates are unchanged. Dotted lines show chance accuracy.