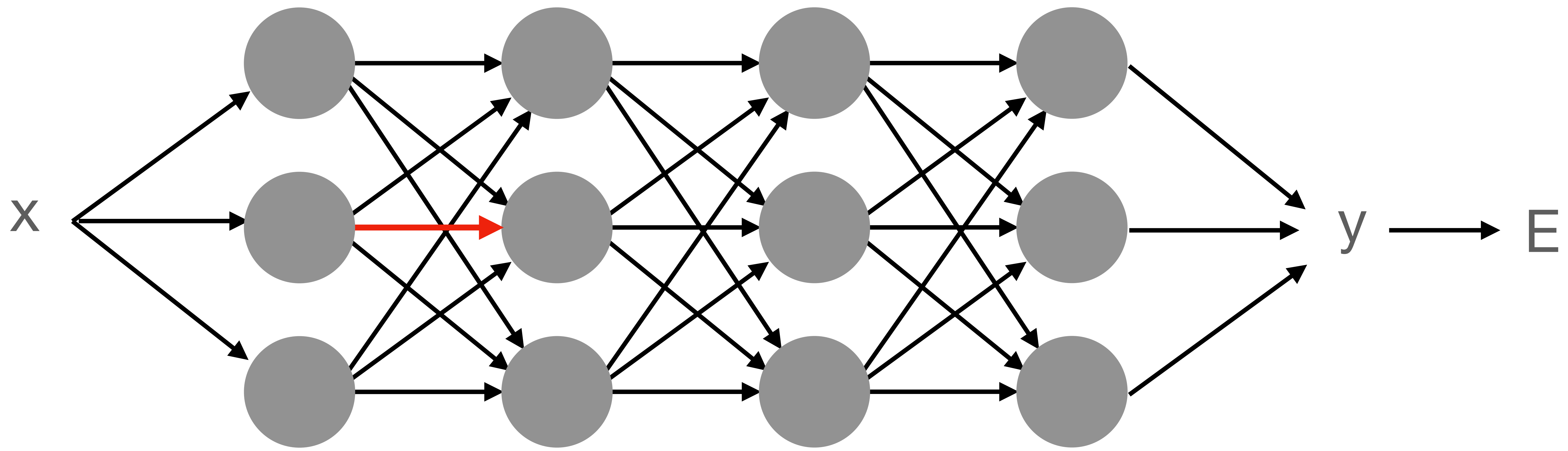
Using Meta-Plasticity to Discover the Biophysics of Learning

Navid Shervani-Tabar, Robert Rosenbaum
University of Notre Dame
Dept. of Applied and Computational Math and Stats

UNIVERSITY OF NOTRE DAME

Learning in Artificial and Biological Neural Networks

$$\Delta w = ?$$



Learning:

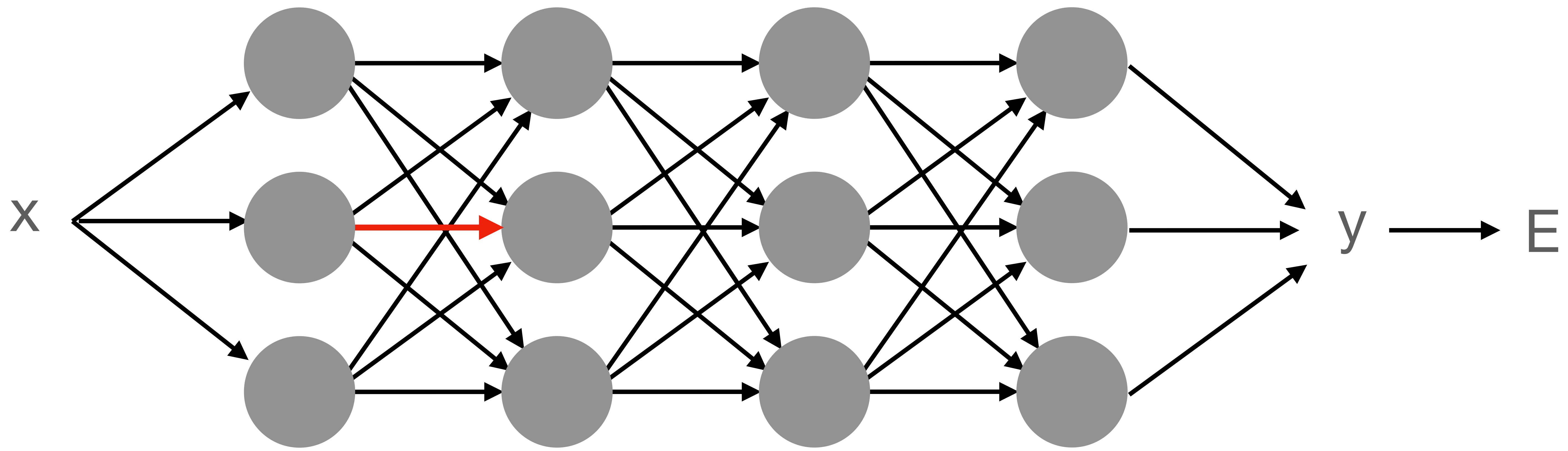
Update the weight in a way that decreases an error function.

Credit Assignment Problem:

How does a single weight contribute to the error?

Learning in Artificial and Biological Neural Networks

gradient descent: $\Delta w \propto -\frac{\partial E}{\partial w}$



Learning:

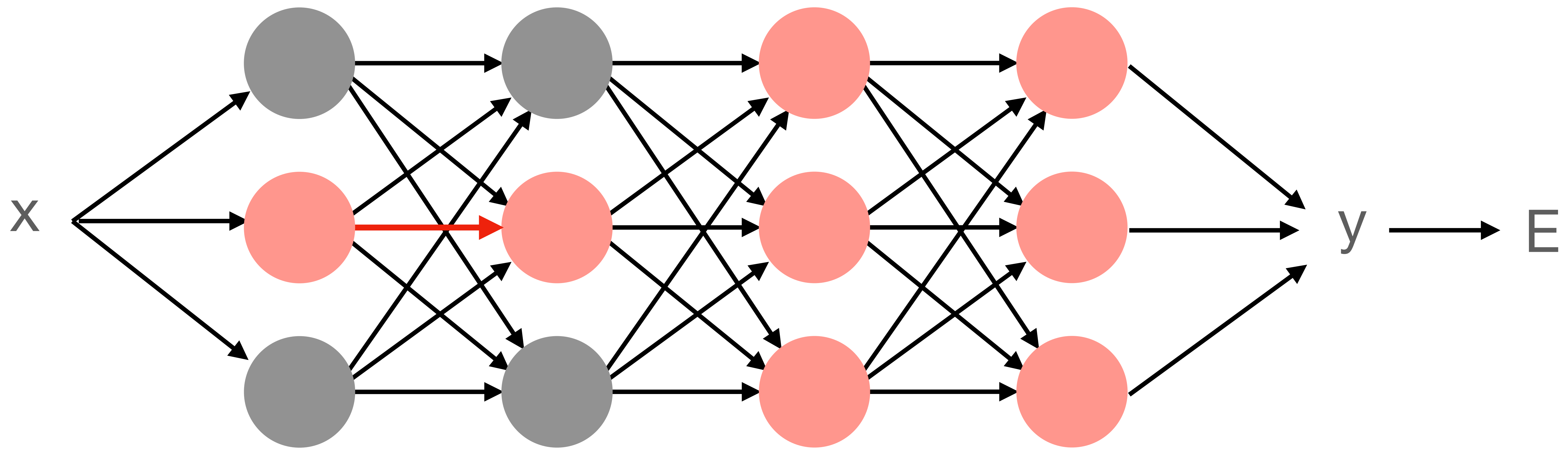
Update the weight in a way that decreases an error function.

Credit Assignment Problem:

How does a single weight contribute to the error?

Learning in Artificial and Biological Neural Networks

gradient descent: $\Delta w \propto -\frac{\partial E}{\partial w}$



Learning:

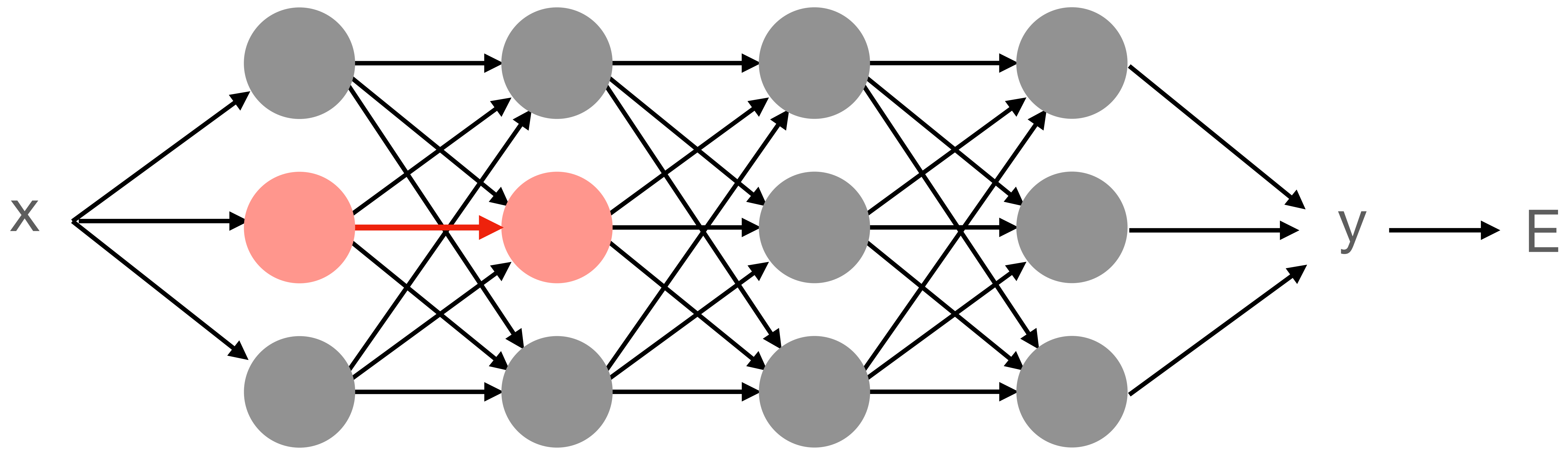
Update the weight in a way that decreases an error function.

Credit Assignment Problem:

How does a single weight contribute to the error?

Learning in Artificial and Biological Neural Networks

local plasticity: $\Delta w \approx F(v_{pre}, v_{post})$



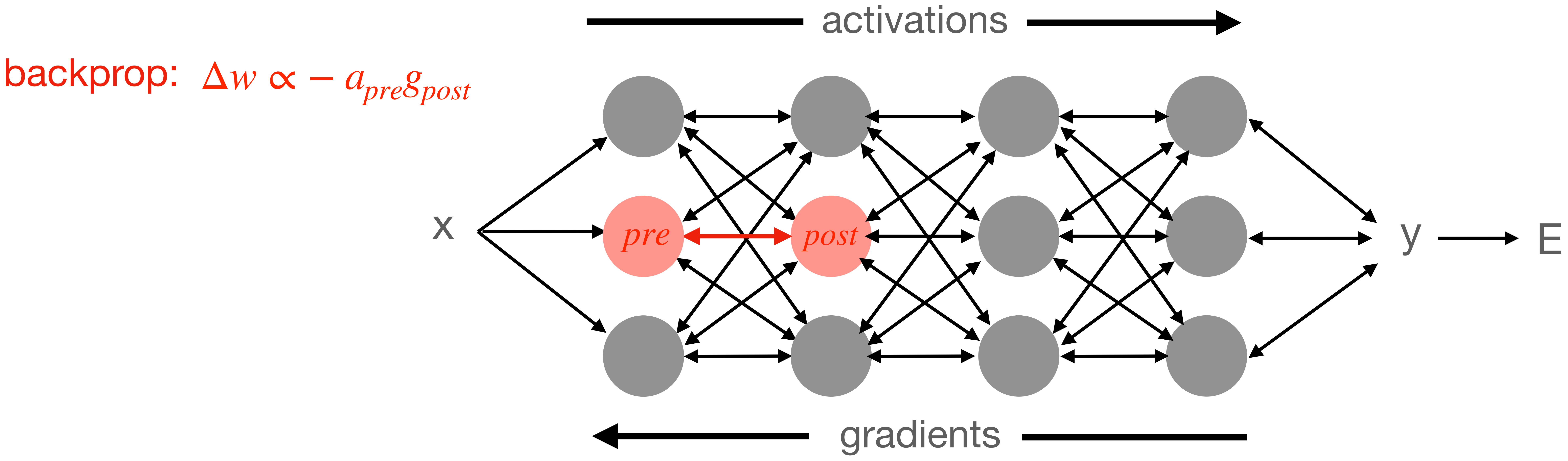
Local Plasticity

Changes to w are largely a function of pre- and post-synaptic activity.

***Biological* Credit Assignment Problem:**

How to achieve effective learning under biologically relevant constraints?

Learning in Artificial and Biological Neural Networks

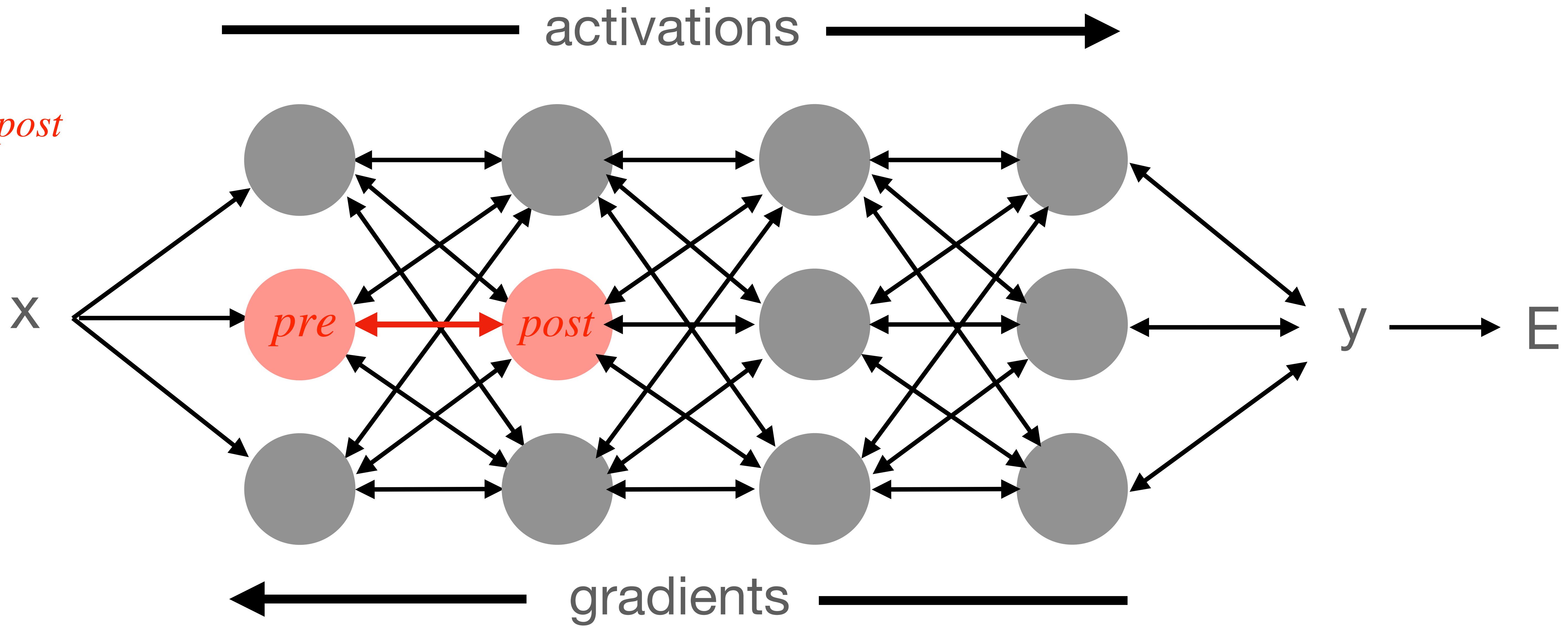


Backpropagation:

Implements gradient descent with local learning rule by passing gradients backward.

Learning in Artificial and Biological Neural Networks

backprop: $\Delta w \propto -a_{pre} g_{post}$



Article | Published: 13 May 2021

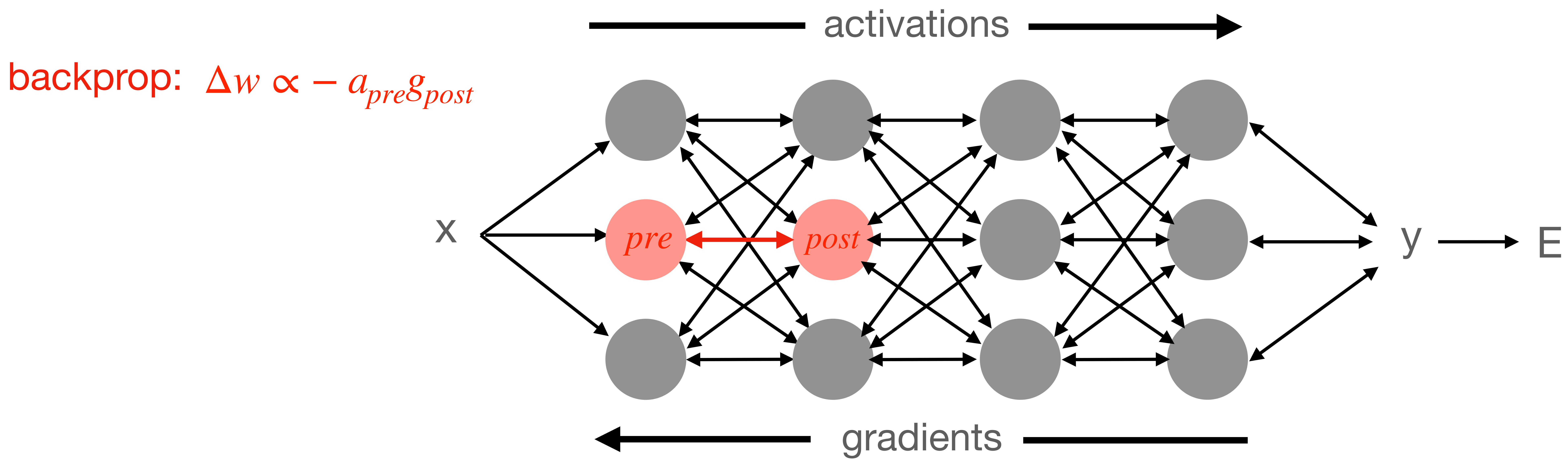
Burst-dependent synaptic plasticity can coordinate learning in hierarchical circuits

Alexandre Payeur, Jordan Guerguiev, Friedemann Zenke, Blake A. Richards & Richard Naud

Nature Neuroscience **24**, 1010–1019 (2021) | Cite this article

activations and gradients separately represented by bursts and spikes?

Learning in Artificial and Biological Neural Networks



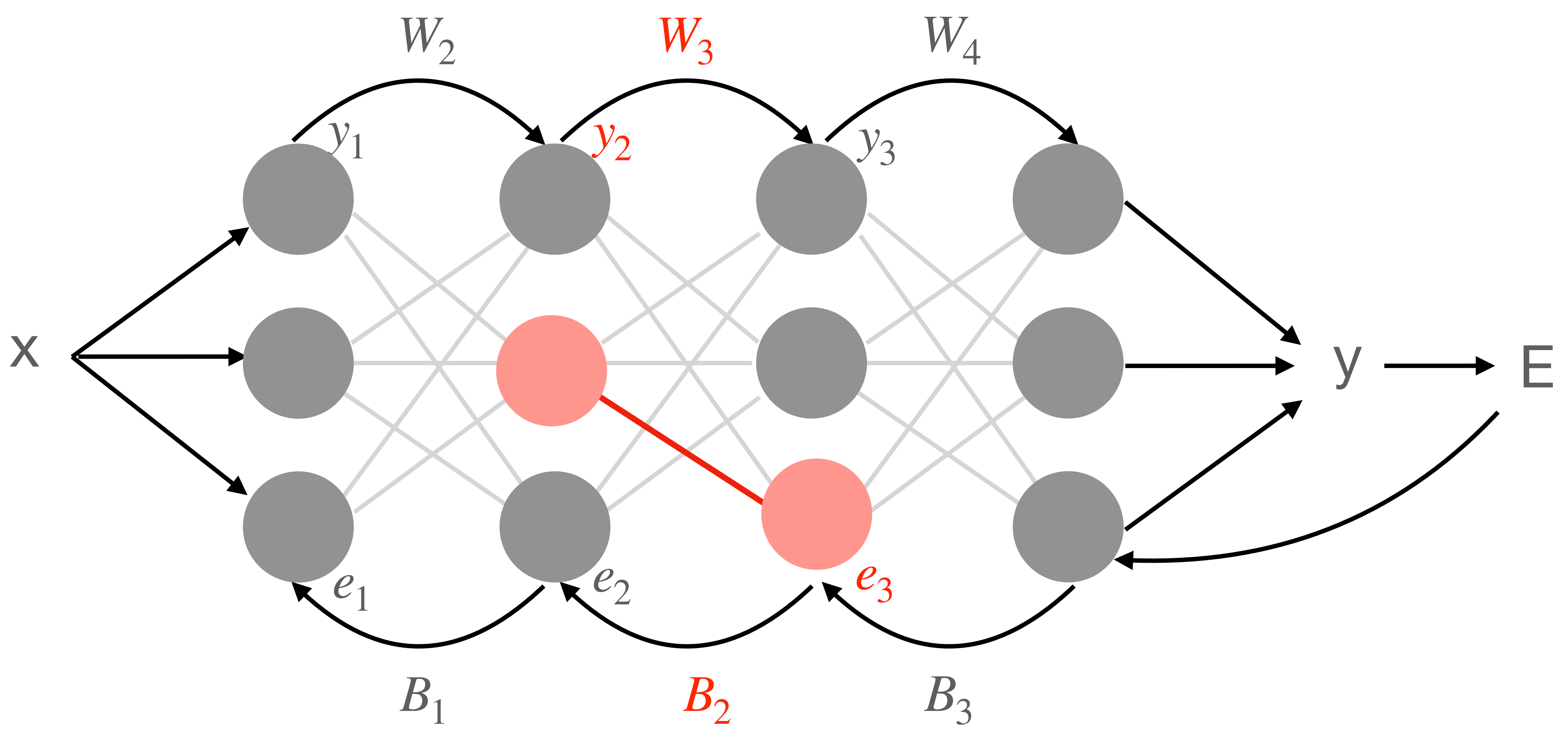
Backpropagation:

Implements gradient descent with local learning rule by passing gradients backward.

Problem:

Symmetric forward and backward connectivity.

Learning in Artificial and Biological Neural Networks



Backprop

$\Delta W_\ell(j,k) = -\eta e_\ell(j) y_{\ell-1}(k)$

$B_\ell = W_{\ell+1}^T$

"weight alignment"

Random feedback alignment

$\Delta W_\ell(j,k) = -\eta e_\ell(j) y_{\ell-1}(k)$

$B_\ell =$ fixed random

(Lillicrap et al., 2016)

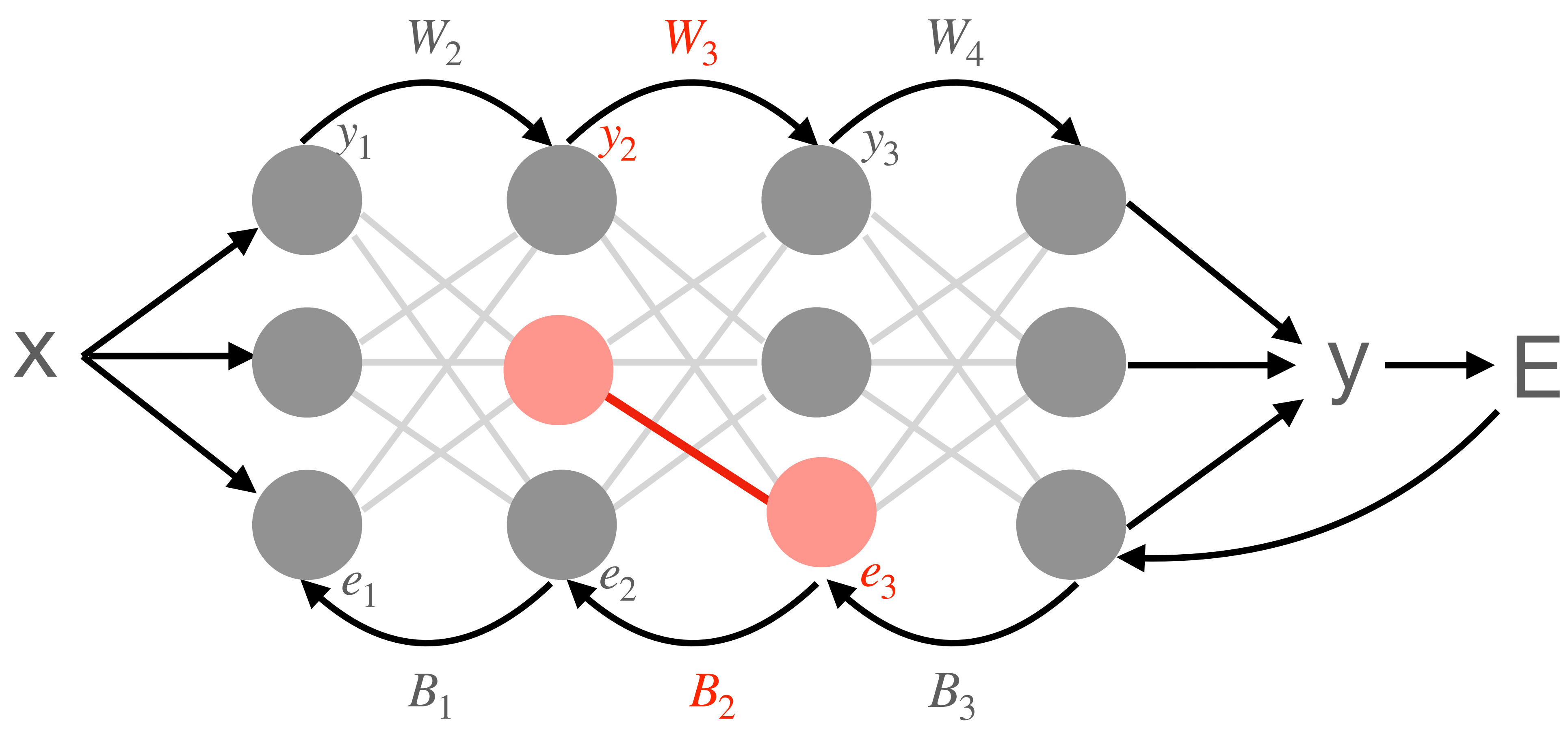
Learning in Artificial and Biological Neural Networks

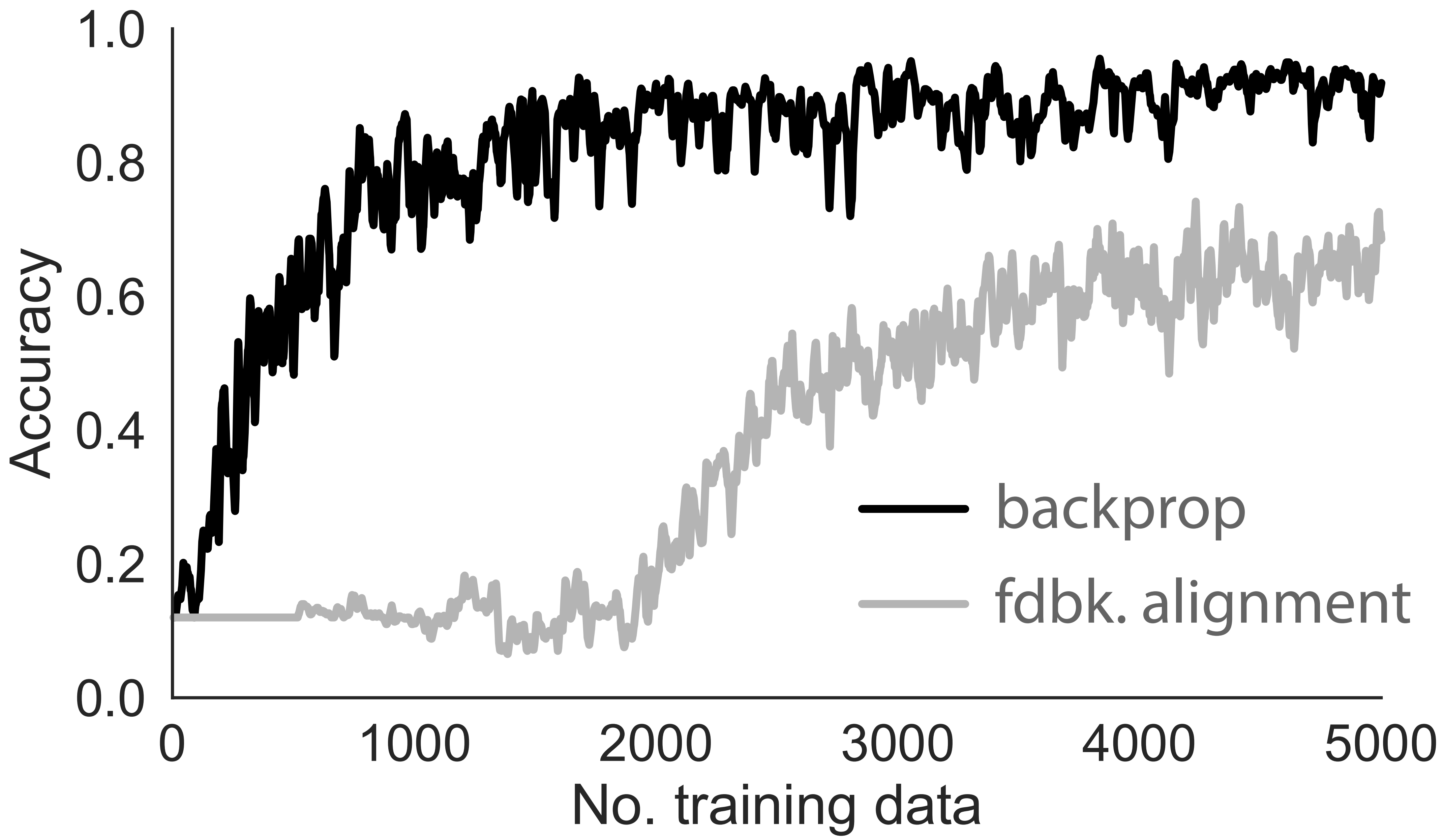
Random feedback alignment

$\Delta W_\ell(j,k) = -\eta e_\ell(j) y_{\ell-1}(k)$

B_ℓ = fixed random

(Lillicrap et al., 2016)



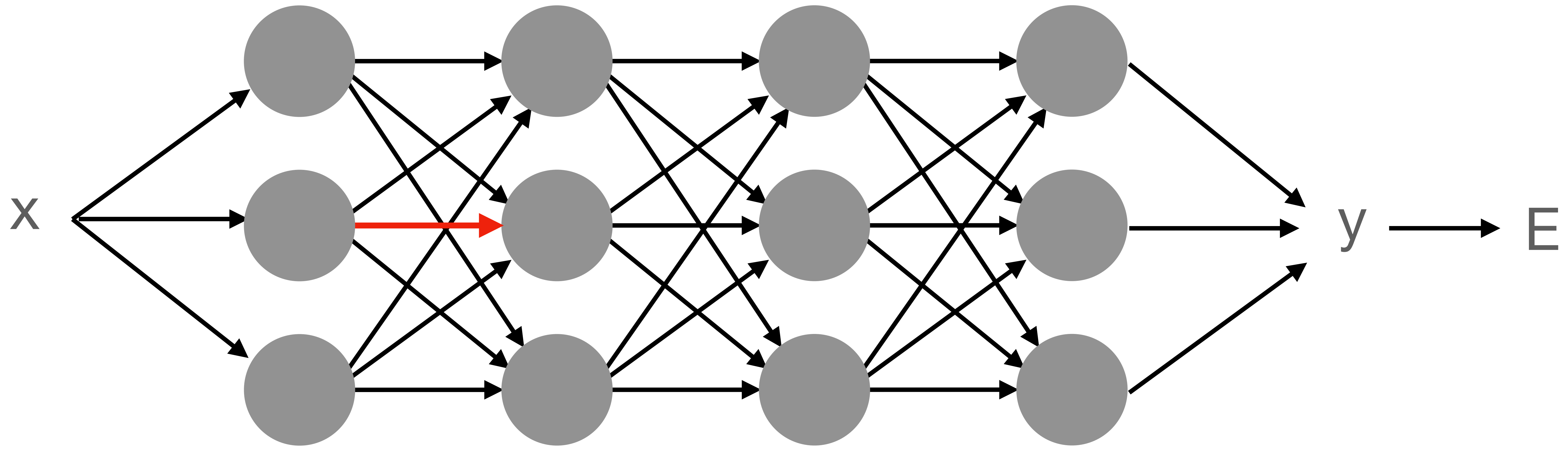


Feedback alignment learns slowly and poorly in deep networks, especially with online learning (batch size =1)

Meta-Plasticity to discover effective plasticity rules

- **Meta-Learning:** Optimize the learning rule across multiple tasks: “***learning to learn.***”
- **Meta-Plasticity:** Meta-learn a plasticity rule.

$$\Delta w = F(\dots ; \theta)$$

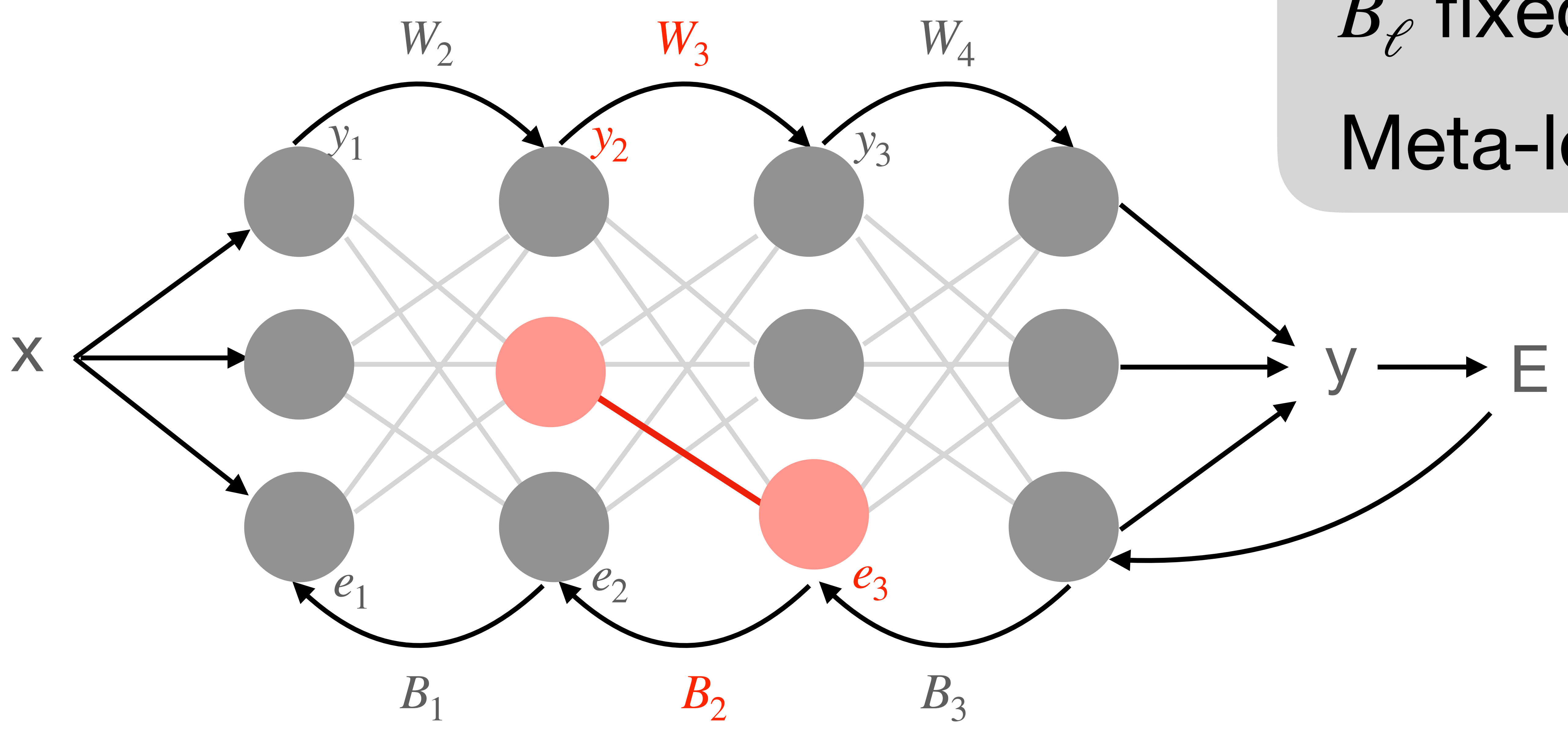


Meta-Plasticity to discover effective plasticity rules

- **Meta-Learning:** Optimize the learning rule across multiple tasks: "***learning to learn.***"
- **Meta-Plasticity:** Meta-learn a plasticity rule.
- Examples:
    - Meta-learn B_ℓ, initial W_ℓ, and parameters for plasticity rule at each synapse (Lindsey and Litwin-Kumar, 2020)
    - Meta-learn parameters for plasticity rule at each synapse. Retain the same W_ℓ across tasks (Miconi et al., 2019)
- **Results:** Better than backprop when generalizing to new tasks.
- **Problem:** Resulting learning rules are ***not interpretable***, difficult to draw biological conclusions.

Our approach to meta-plasticity

- **Meta-parameter sharing**
  - All synapses share the same local plasticity rule.
  - Produces a single, ***interpretable*** plasticity rule.



$$\Delta W_\ell(j,k) = F(y_{pre}(k), y_{post}(j), e_{pre}(k), e_{post}(j); \theta)$$

B_ℓ fixed, random

Meta-learn θ

(See also: Confavreux et al., NeurIPS, 2020)

Our approach to meta-plasticity

$\Delta W_\ell(j,k) = F(y_{pre}(k), y_{post}(j), e_{pre}(k), e_{post}(j); \theta)$

B_ℓ fixed, random

Meta-learn θ

outer loop over tasks (“episodes”)

for each task:

Initialize W’s and B’s

for each data point:

Forward and backward pass to compute y’s and e’s

Update $W's$ using $\Delta W_\ell = F(\ \dots\ ;\ \theta)$

Compute outputs $\hat{Y}$ on unseen “query” data

Update theta using gradient of meta-cost: $J^{meta} = L(\hat{Y}, Y) + \lambda ||\theta||_1$

Our approach to meta-plasticity

$\Delta W_\ell(j,k) = F(y_{pre}(k), y_{post}(j), e_{pre}(k), e_{post}(j); \theta)$

B_ℓ fixed, random

Meta-learn θ

for each task:

Initialize W's and B's

inner loop

for each data point:

Forward and backward pass to compute y's and e's

Update $W's$ using $\Delta W_\ell = F(\ \dots\ ;\ \theta)$

Compute outputs $\hat{Y}$ on unseen "query" data

Update theta using gradient of meta-cost: $J^{meta} = L(\hat{Y}, Y) + \lambda||\theta||_1$

Our approach to meta-plasticity

$\Delta W_\ell(j,k) = F(y_{pre}(k), y_{post}(j), e_{pre}(k), e_{post}(j); \theta)$

B_ℓ fixed, random

Meta-learn θ

for each task:

Initialize W's and B's inner loop learns from scratch

for each data point:

Forward and backward pass to compute y's and e's

Update $W's$ using $\Delta W_\ell = F(\ldots ; \theta)$

Compute outputs $\hat{Y}$ on unseen "query" data

Update theta using gradient of meta-cost: $J^{meta} = L(\hat{Y}, Y) + \lambda ||\theta||_1$

Our approach to meta-plasticity

$\Delta W_\ell(j,k) = F(y_{pre}(k), y_{post}(j), e_{pre}(k), e_{post}(j); \theta)$

B_ℓ fixed, random

Meta-learn θ

for each task:

Initialize W's and B's

for each data point: online learning (batch size 1)

Forward and backward pass to compute y's and e's

Update $W's$ using $\Delta W_\ell = F(\dots ; \theta)$

Compute outputs $\hat{Y}$ on unseen "query" data

Update theta using gradient of meta-cost: $J^{meta} = L(\hat{Y}, Y) + \lambda ||\theta||_1$

Our approach to meta-plasticity

$\Delta W_\ell(j,k) = F(y_{pre}(k), y_{post}(j), e_{pre}(k), e_{post}(j); \theta)$

B_ℓ fixed, random

Meta-learn θ

for each task:

Initialize W's and B's

for each data point:

Forward and backward pass to compute y's and e's

Update $W's$ using $\Delta W_\ell = F(\ \dots\ ;\ \theta)$

Compute outputs $\hat{Y}$ on unseen "query" data

Update theta using gradient of meta-cost: $J^{meta} = L(\hat{Y}, Y) + \lambda ||\theta||_1$

L1 meta-regularization
promotes simple plasticity rules

Our approach to meta-plasticity

$\Delta W_\ell(j,k) = F(y_{pre}(k), y_{post}(j), e_{pre}(k), e_{post}(j); \theta)$

B_ℓ fixed, random

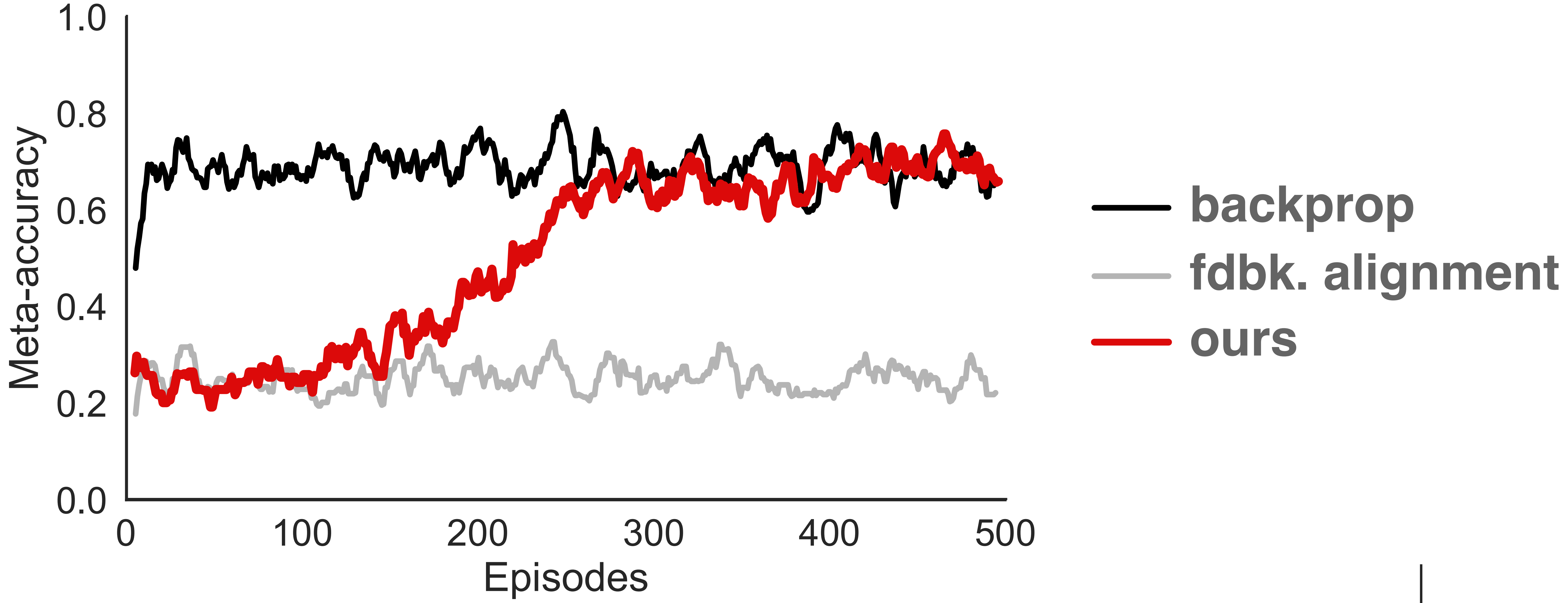
Meta-learn θ

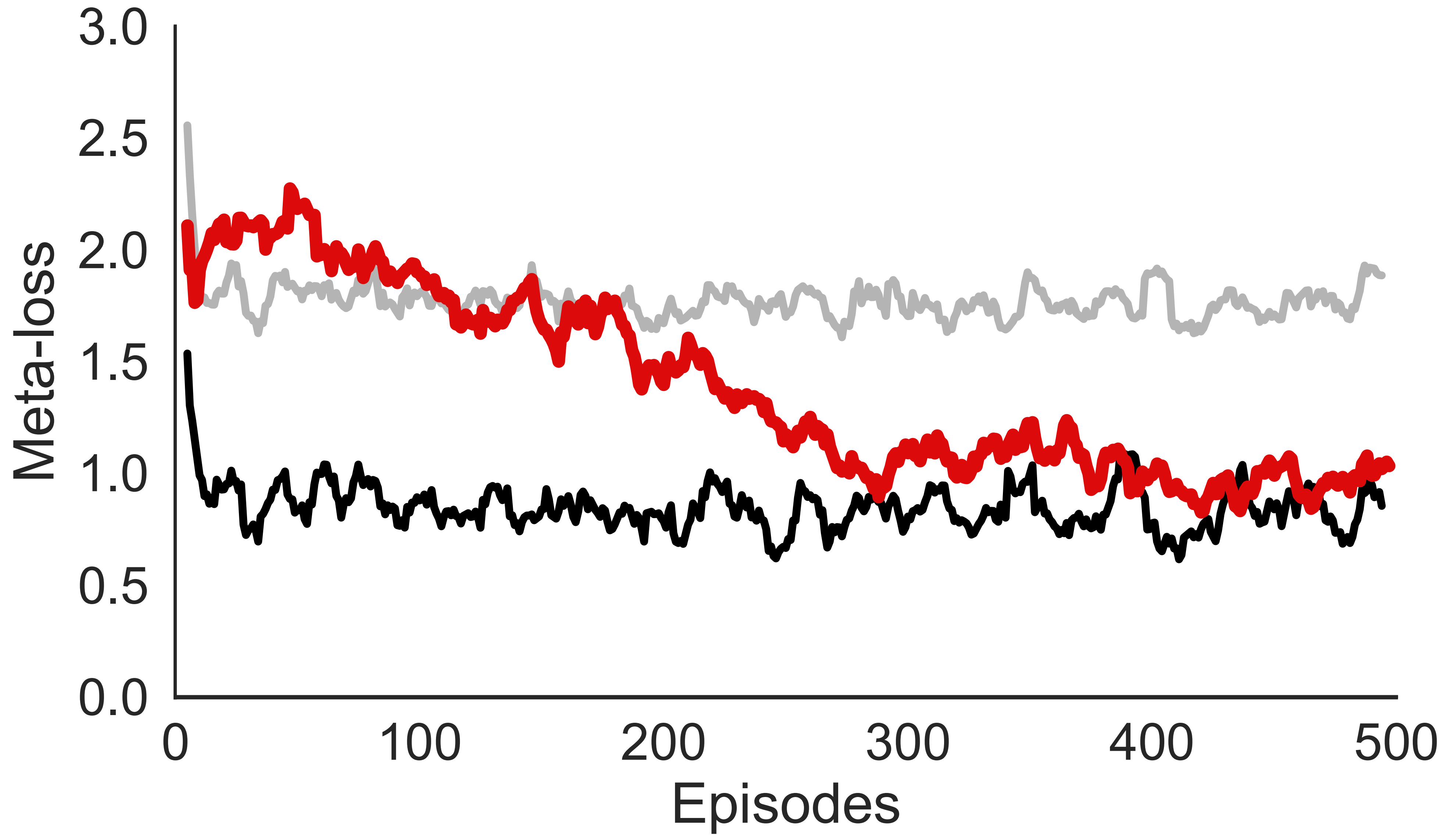
- 5-layer fully connected perceptron
- Each task is a subset of EMNIST
- Inner loop data set size = 256
- Linear combo of 10 bio-inspired plasticity rules:

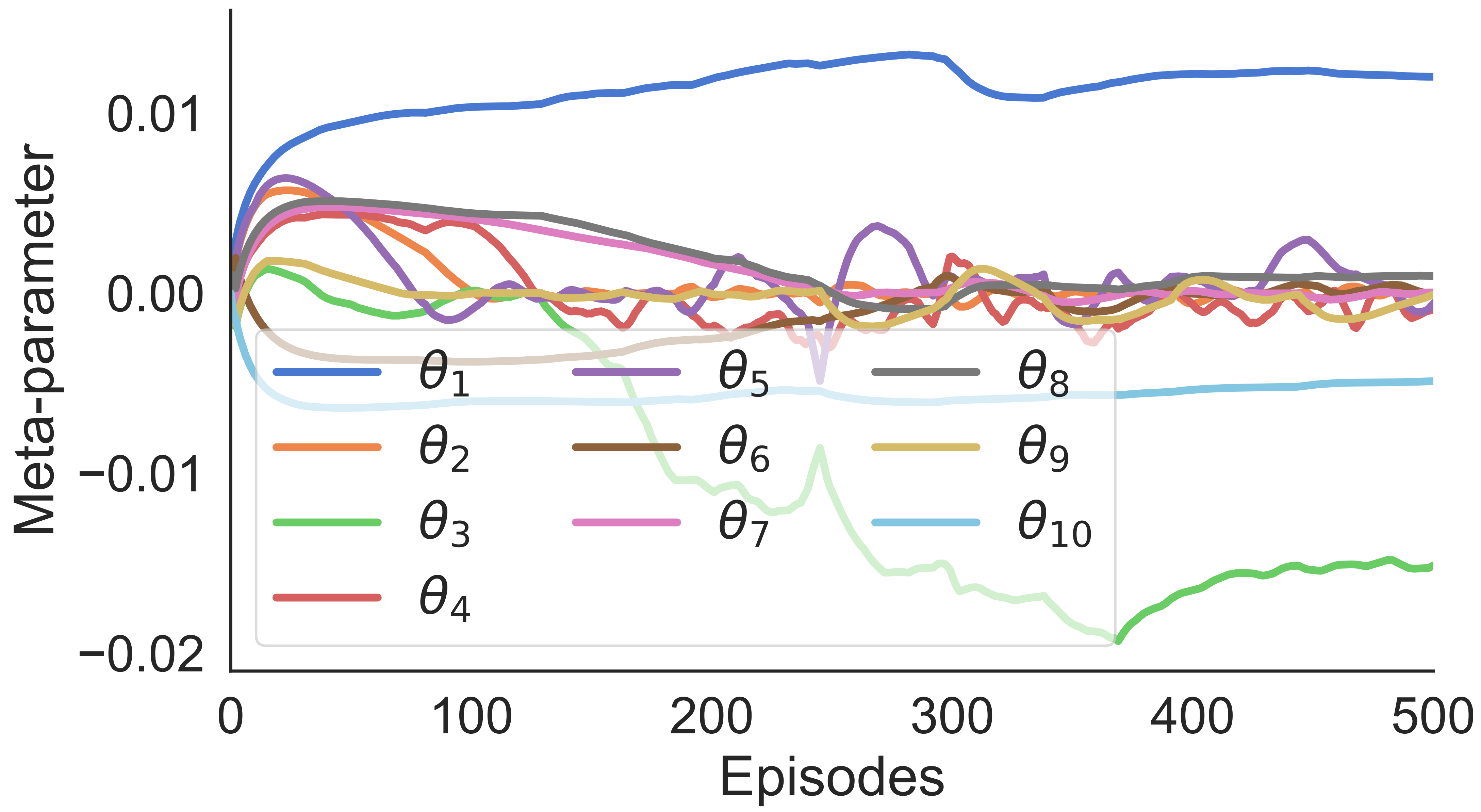
$$F(\dots;\theta) = \sum_k \theta_k \mathcal{F}^{(k)}(\dots)$$

$$\begin{aligned}
\mathcal{F}^{(1)} &= -\theta_1 \boldsymbol{e}_\ell \boldsymbol{y}_{\ell-1}^T,\\
\mathcal{F}^{(2)} &= -\theta_2 \boldsymbol{y}_\ell \boldsymbol{e}_{\ell-1}^T,\\
\mathcal{F}^{(3)} &= -\theta_3 \boldsymbol{e}_\ell \boldsymbol{e}_{\ell-1}^T,\\
\mathcal{F}^{(4)} &= -\theta_4 \boldsymbol{W}_{\ell-1,\ell},\\
\mathcal{F}^{(5)} &= -\theta_5 \boldsymbol{1}_\ell \boldsymbol{e}_{\ell-1}^T,\\
\mathcal{F}^{(6)} &= -\theta_6 \boldsymbol{e}_\ell \boldsymbol{1}_\ell^T \boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T,\\
\mathcal{F}^{(7)} &= -\theta_7 \boldsymbol{y}_\ell \boldsymbol{y}_\ell^T \boldsymbol{W}_{\ell-1,\ell} \boldsymbol{e}_{\ell-1} \boldsymbol{e}_{\ell-1}^T,\\
\mathcal{F}^{(8)} &= -\theta_8 \boldsymbol{e}_\ell \boldsymbol{y}_\ell^T \boldsymbol{W}_{\ell-1,\ell} \boldsymbol{e}_{\ell-1} \boldsymbol{y}_{\ell-1}^T,\\
\mathcal{F}^{(9)} &= -\theta_9 \boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T \boldsymbol{W}_{\ell-1,\ell}^T \boldsymbol{e}_\ell \boldsymbol{e}_{\ell-1}^T,\\
\mathcal{F}^{(10)} &= -\theta_{10}(\boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T - (\boldsymbol{y}_\ell \boldsymbol{y}_\ell^T)\boldsymbol{W}_{\ell-1,\ell})
\end{aligned}$$

Results


1.0
0.8
0.6
0.4
0.2
0.0
Meta-accuracy
0
100
200
300
400
500
Episodes
backprop
fdbk. alignment
ours


3.0
2.5
2.0
1.5
1.0
0.5
0.0
Meta-loss
0
100
200
300
400
500
Episodes


0.01
0.00
−0.01
−0.02
Meta-parameter
0
100
200
300
400
500
Episodes
θ_1
θ_2
θ_3
θ_4
θ_5
θ_6
θ_7
θ_8
θ_9
θ_{10}

Results

$$\boxed{\mathcal{F}^{(1)} = -\theta_1 \boldsymbol{e}_\ell \boldsymbol{y}_{\ell-1}^T,}$$

$$\mathcal{F}^{(2)} = -\theta_2 \boldsymbol{y}_\ell \boldsymbol{e}_{\ell-1}^T,$$

$$\boxed{\mathcal{F}^{(3)} = -\theta_3 \boldsymbol{e}_\ell \boldsymbol{e}_{\ell-1}^T,}$$

$$\mathcal{F}^{(4)} = -\theta_4 \boldsymbol{W}_{\ell-1,\ell},$$

$$\mathcal{F}^{(5)} = -\theta_5 \mathbf{1}_\ell \boldsymbol{e}_{\ell-1}^T,$$

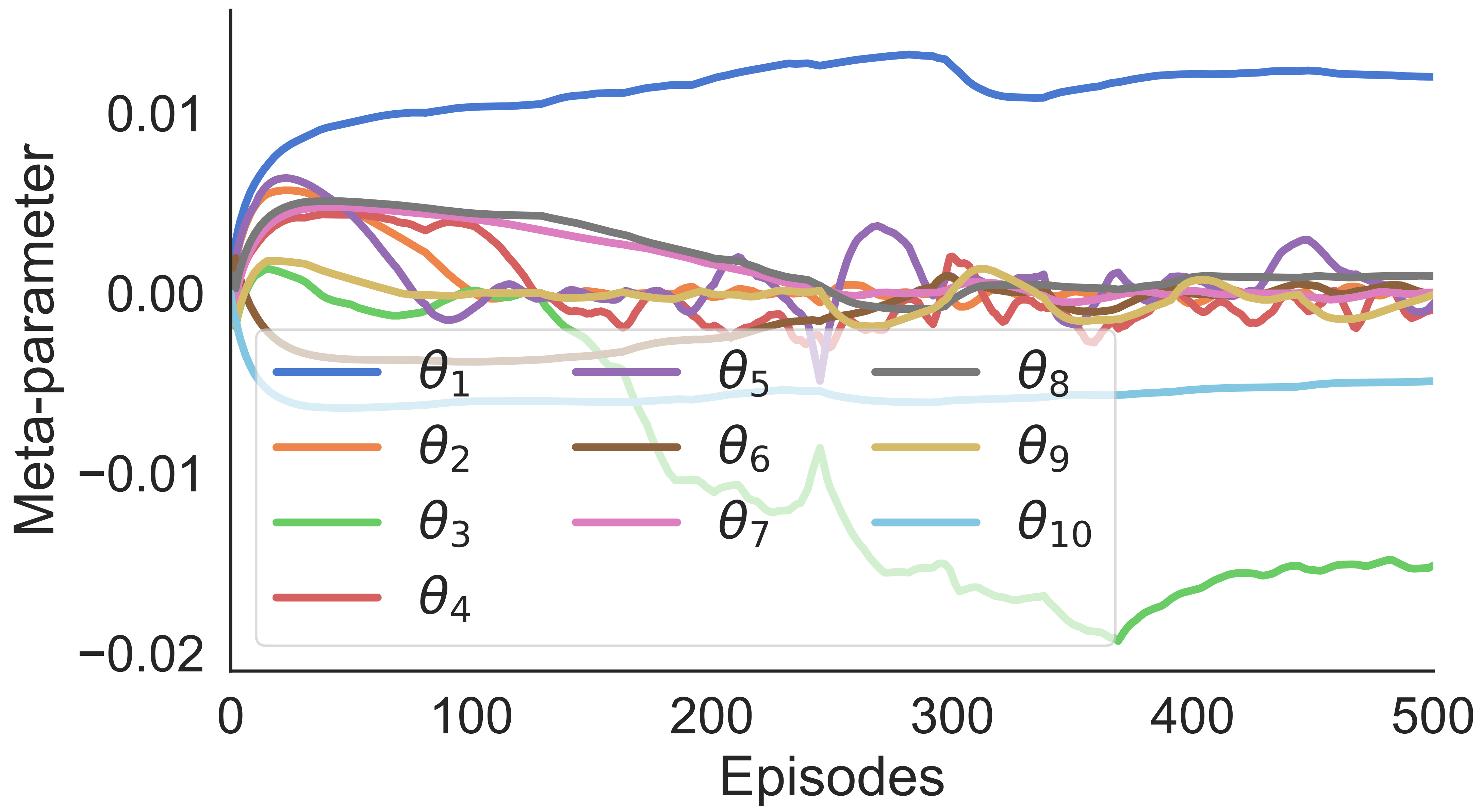
$$\mathcal{F}^{(6)} = -\theta_6 \boldsymbol{e}_\ell \mathbf{1}_\ell^T \boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T,$$

$$\mathcal{F}^{(7)} = -\theta_7 \boldsymbol{y}_\ell \boldsymbol{y}_\ell^T \boldsymbol{W}_{\ell-1,\ell} \boldsymbol{e}_{\ell-1} \boldsymbol{e}_{\ell-1}^T,$$

$$\mathcal{F}^{(8)} = -\theta_8 \boldsymbol{e}_\ell \boldsymbol{y}_\ell^T \boldsymbol{W}_{\ell-1,\ell} \boldsymbol{e}_{\ell-1} \boldsymbol{y}_{\ell-1}^T,$$

$$\mathcal{F}^{(9)} = -\theta_9 \boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T \boldsymbol{W}_{\ell-1,\ell}^T \boldsymbol{e}_\ell \boldsymbol{e}_{\ell-1}^T,$$

$$\boxed{\mathcal{F}^{(10)} = -\theta_{10}(\boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T - (\boldsymbol{y}_\ell \boldsymbol{y}_\ell^T)\boldsymbol{W}_{\ell-1,\ell})}$$



Results

$$\boxed{\mathcal{F}^{(1)} = -\theta_1 \boldsymbol{e}_\ell \boldsymbol{y}_{\ell-1}^T,}$$

$$\mathcal{F}^{(2)} = -\theta_2 \boldsymbol{y}_\ell \boldsymbol{e}_{\ell-1}^T,$$

$$\boxed{\mathcal{F}^{(3)} = -\theta_3 \boldsymbol{e}_\ell \boldsymbol{e}_{\ell-1}^T,}$$

$$\mathcal{F}^{(4)} = -\theta_4 \boldsymbol{W}_{\ell-1,\ell},$$

$$\mathcal{F}^{(5)} = -\theta_5 \boldsymbol{1}_\ell \boldsymbol{e}_{\ell-1}^T,$$

$$\mathcal{F}^{(6)} = -\theta_6 \boldsymbol{e}_\ell \boldsymbol{1}_\ell^T \boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T,$$

$$\mathcal{F}^{(7)} = -\theta_7 \boldsymbol{y}_\ell \boldsymbol{y}_\ell^T \boldsymbol{W}_{\ell-1,\ell} \boldsymbol{e}_{\ell-1} \boldsymbol{e}_{\ell-1}^T,$$

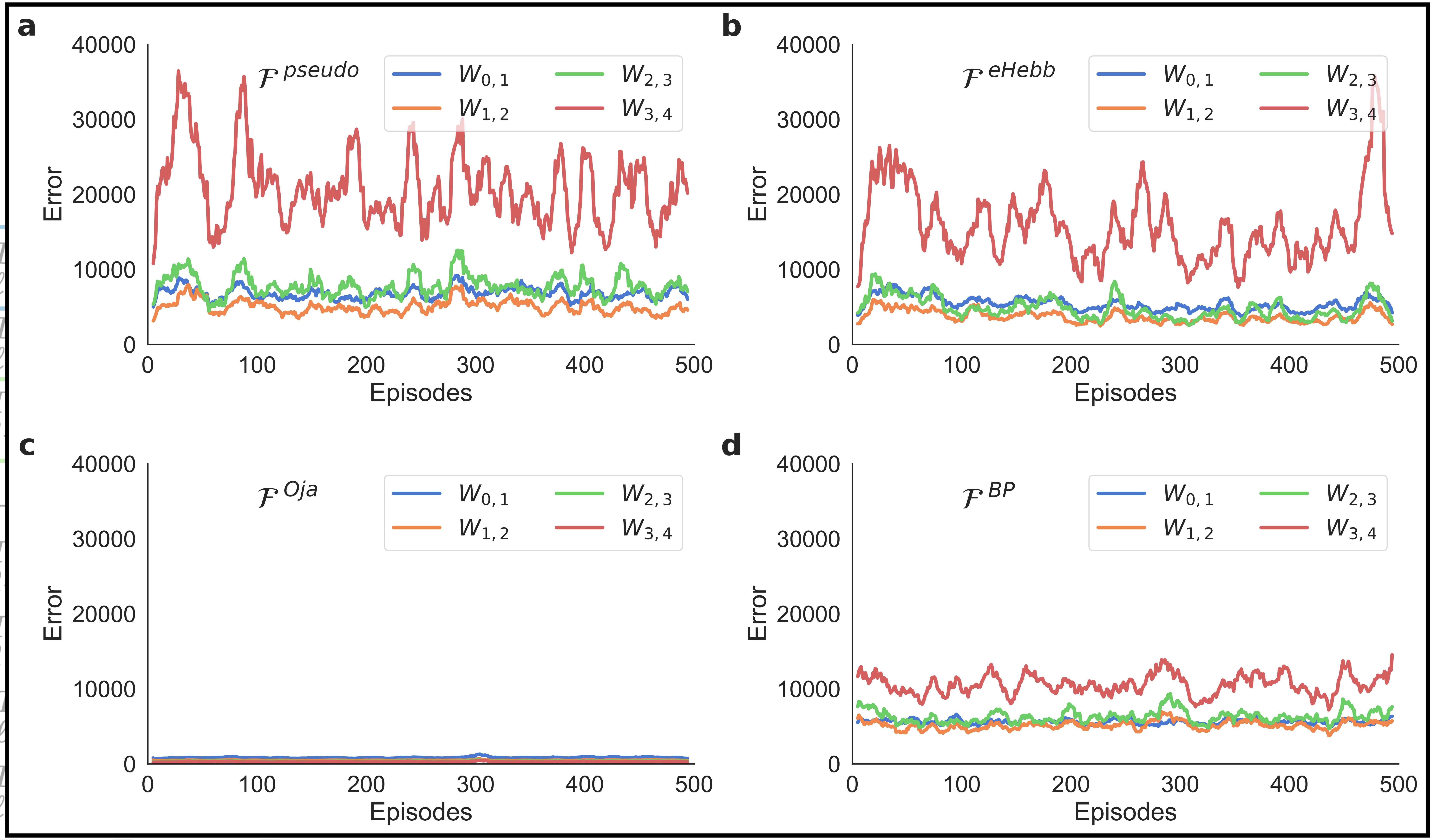
$$\mathcal{F}^{(8)} = -\theta_8 \boldsymbol{e}_\ell \boldsymbol{y}_\ell^T \boldsymbol{W}_{\ell-1,\ell} \boldsymbol{e}_{\ell-1} \boldsymbol{y}_{\ell-1}^T,$$

$$\mathcal{F}^{(9)} = -\theta_9 \boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T \boldsymbol{W}_{\ell-1,\ell}^T \boldsymbol{e}_\ell \boldsymbol{e}_{\ell-1}^T,$$

$$\boxed{\mathcal{F}^{(10)} = -\theta_{10} (\boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T - (\boldsymbol{y}_\ell \boldsymbol{y}_\ell^T) \boldsymbol{W}_{\ell-1,\ell})}$$

Oja's rule on activations.
Causes hidden layers to perform
PCA-like orthonormalization.

Results



$\mathcal{F}^{(1)} = -\theta_1 \boldsymbol{e}_\ell \boldsymbol{y}_\ell^T$

$\mathcal{F}^{(2)} = -\theta_2 \boldsymbol{y}_\ell \boldsymbol{e}_\ell^T$

$\mathcal{F}^{(3)} = -\theta_3 \boldsymbol{e}_\ell \boldsymbol{e}_\ell^T$

$\mathcal{F}^{(4)} = -\theta_4 \boldsymbol{W}_{\ell-}$

$\mathcal{F}^{(5)} = -\theta_5 \boldsymbol{1}_\ell \boldsymbol{e}_\ell^T$

$\mathcal{F}^{(6)} = -\theta_6 \boldsymbol{e}_\ell \boldsymbol{1}_\ell^T$

$\mathcal{F}^{(7)} = -\theta_7 \boldsymbol{y}_\ell \boldsymbol{y}_\ell^T$

$\mathcal{F}^{(8)} = -\theta_8 \boldsymbol{e}_\ell \boldsymbol{y}_\ell^T$

$$\mathcal{F}^{(9)} = -\theta_9 \boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T \boldsymbol{W}_{\ell-1,\ell}^T \boldsymbol{e}_\ell \boldsymbol{e}_{\ell-1}^T,$$

$$\mathcal{F}^{(10)} = -\theta_{10}(\boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T - (\boldsymbol{y}_\ell \boldsymbol{y}_\ell^T)\boldsymbol{W}_{\ell-1,\ell})$$

Oja's rule on activations.
Causes hidden layers to perform
PCA-like orthonormalization.

Results

$$\boxed{\mathcal{F}^{(1)} = -\theta_1 \boldsymbol{e}_\ell \boldsymbol{y}_{\ell-1}^T,}$$

Pseudo-gradient term. If $W_\ell = B_{\ell-1}^T$, this term would implement backprop.

$$\mathcal{F}^{(2)} = -\theta_2 \boldsymbol{y}_\ell \boldsymbol{e}_{\ell-1}^T,$$

$$\boxed{\mathcal{F}^{(3)} = -\theta_3 \boldsymbol{e}_\ell \boldsymbol{e}_{\ell-1}^T,}$$

$$\mathcal{F}^{(4)} = -\theta_4 \boldsymbol{W}_{\ell-1,\ell},$$

$$\mathcal{F}^{(5)} = -\theta_5 \boldsymbol{1}_\ell \boldsymbol{e}_{\ell-1}^T,$$

$$\mathcal{F}^{(6)} = -\theta_6 \boldsymbol{e}_\ell \boldsymbol{1}_\ell^T \boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T,$$

$$\mathcal{F}^{(7)} = -\theta_7 \boldsymbol{y}_\ell \boldsymbol{y}_\ell^T \boldsymbol{W}_{\ell-1,\ell} \boldsymbol{e}_{\ell-1} \boldsymbol{e}_{\ell-1}^T,$$

$$\mathcal{F}^{(8)} = -\theta_8 \boldsymbol{e}_\ell \boldsymbol{y}_\ell^T \boldsymbol{W}_{\ell-1,\ell} \boldsymbol{e}_{\ell-1} \boldsymbol{y}_{\ell-1}^T,$$

$$\mathcal{F}^{(9)} = -\theta_9 \boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T \boldsymbol{W}_{\ell-1,\ell}^T \boldsymbol{e}_\ell \boldsymbol{e}_{\ell-1}^T,$$

$$\boxed{\mathcal{F}^{(10)} = -\theta_{10}(\boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T - (\boldsymbol{y}_\ell \boldsymbol{y}_\ell^T) \boldsymbol{W}_{\ell-1,\ell})}$$

Oja's rule on activations.
Causes hidden layers to perform PCA-like orthonormalization.

Results

$$\boxed{\mathcal{F}^{(1)} = -\theta_1 \boldsymbol{e}_\ell \boldsymbol{y}_{\ell-1}^T,}$$

$$\mathcal{F}^{(2)} = -\theta_2 \boldsymbol{y}_\ell \boldsymbol{e}_{\ell-1}^T,$$

$$\boxed{\mathcal{F}^{(3)} = -\theta_3 \boldsymbol{e}_\ell \boldsymbol{e}_{\ell-1}^T,}$$

$$\mathcal{F}^{(4)} = -\theta_4 \boldsymbol{W}_{\ell-1,\ell},$$

$$\mathcal{F}^{(5)} = -\theta_5 \boldsymbol{1}_\ell \boldsymbol{e}_{\ell-1}^T,$$

$$\mathcal{F}^{(6)} = -\theta_6 \boldsymbol{e}_\ell \boldsymbol{1}_\ell^T \boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T,$$

$$\mathcal{F}^{(7)} = -\theta_7 \boldsymbol{y}_\ell \boldsymbol{y}_\ell^T \boldsymbol{W}_{\ell-1,\ell} \boldsymbol{e}_{\ell-1} \boldsymbol{e}_{\ell-1}^T,$$

$$\mathcal{F}^{(8)} = -\theta_8 \boldsymbol{e}_\ell \boldsymbol{y}_\ell^T \boldsymbol{W}_{\ell-1,\ell} \boldsymbol{e}_{\ell-1} \boldsymbol{y}_{\ell-1}^T,$$

$$\mathcal{F}^{(9)} = -\theta_9 \boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T \boldsymbol{W}_{\ell-1,\ell}^T \boldsymbol{e}_\ell \boldsymbol{e}_{\ell-1}^T,$$

$$\boxed{\mathcal{F}^{(10)} = -\theta_{10} (\boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T - (\boldsymbol{y}_\ell \boldsymbol{y}_\ell^T) \boldsymbol{W}_{\ell-1,\ell})}$$

Pseudo-gradient term. If $W_\ell = B_{\ell-1}^T$, this term would implement backprop.

Hebbian plasticity on errors.
Theorem: In a linear network, this term pushes W_ℓ toward $B_{\ell-1}^T$

Oja's rule on activations.
Causes hidden layers to perform PCA-like orthonormalization.

Results

$$\boxed{\mathcal{F}^{(1)} = -\theta_1 \boldsymbol{e}_\ell \boldsymbol{y}_{\ell-1}^T,}$$
$$\mathcal{F}^{(2)} = -\theta_2 \boldsymbol{y}_\ell \boldsymbol{e}_{\ell-1}^T,$$
$$\boxed{\mathcal{F}^{(3)} = -\theta_3 \boldsymbol{e}_\ell \boldsymbol{e}_{\ell-1}^T,}$$
$$\mathcal{F}^{(4)} = -\theta_4 \boldsymbol{W}_{\ell-1,\ell},$$
$$\mathcal{F}^{(5)} = -\theta_5 \boldsymbol{1}_\ell \boldsymbol{e}_{\ell-1}^T,$$
$$\mathcal{F}^{(6)} = -\theta_6 \boldsymbol{e}_\ell \boldsymbol{1}_\ell^T \boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T,$$
$$\mathcal{F}^{(7)} = -\theta_7 \boldsymbol{y}_\ell \boldsymbol{y}_\ell^T \boldsymbol{W}_{\ell-1,\ell} \boldsymbol{e}_{\ell-1} \boldsymbol{e}_{\ell-1}^T,$$
$$\mathcal{F}^{(8)} = -\theta_8 \boldsymbol{e}_\ell \boldsymbol{y}_\ell^T \boldsymbol{W}_{\ell-1,\ell} \boldsymbol{e}_{\ell-1} \boldsymbol{y}_{\ell-1}^T,$$
$$\mathcal{F}^{(9)} = -\theta_9 \boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T \boldsymbol{W}_{\ell-1,\ell}^T \boldsymbol{e}_\ell \boldsymbol{e}_{\ell-1}^T,$$
$$\boxed{\mathcal{F}^{(10)} = -\theta_{10}(\boldsymbol{y}_\ell \boldsymbol{y}_{\ell-1}^T - (\boldsymbol{y}_\ell \boldsymbol{y}_\ell^T) \boldsymbol{W}_{\ell-1,\ell})}$$

Hebbian plasticity on errors.
Theorem: In a linear network, this term pushes W_ℓ toward $B_{\ell-1}^T$

Could this rule help align weights for learning via burst-dependent plasticity?

Article | Published: 13 May 2021

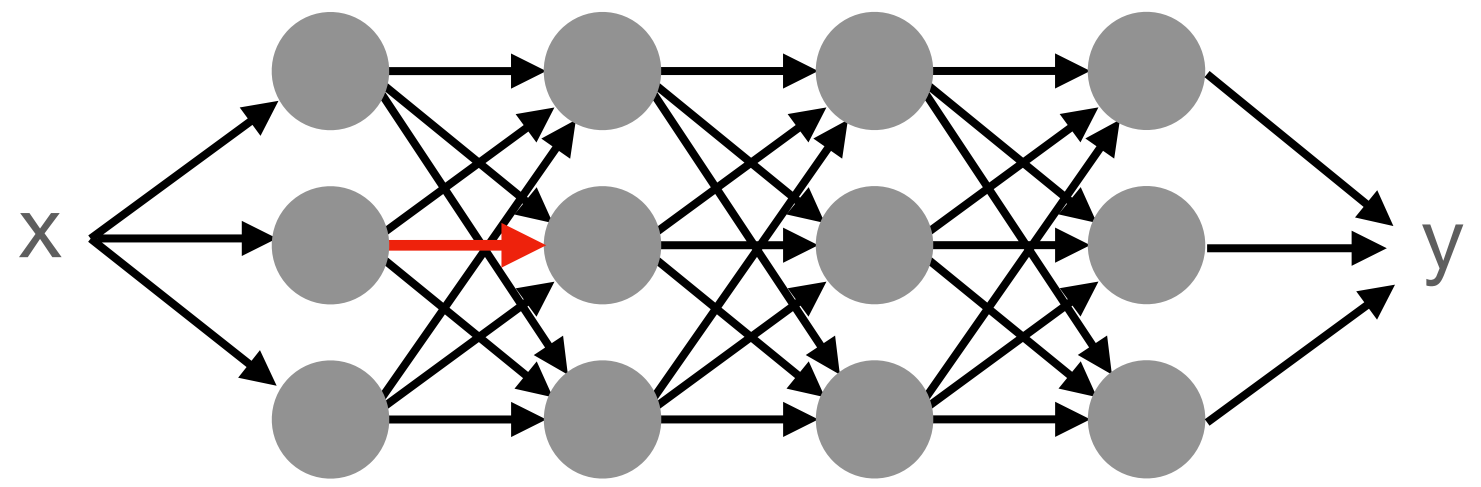
Burst-dependent synaptic plasticity can coordinate learning in hierarchical circuits

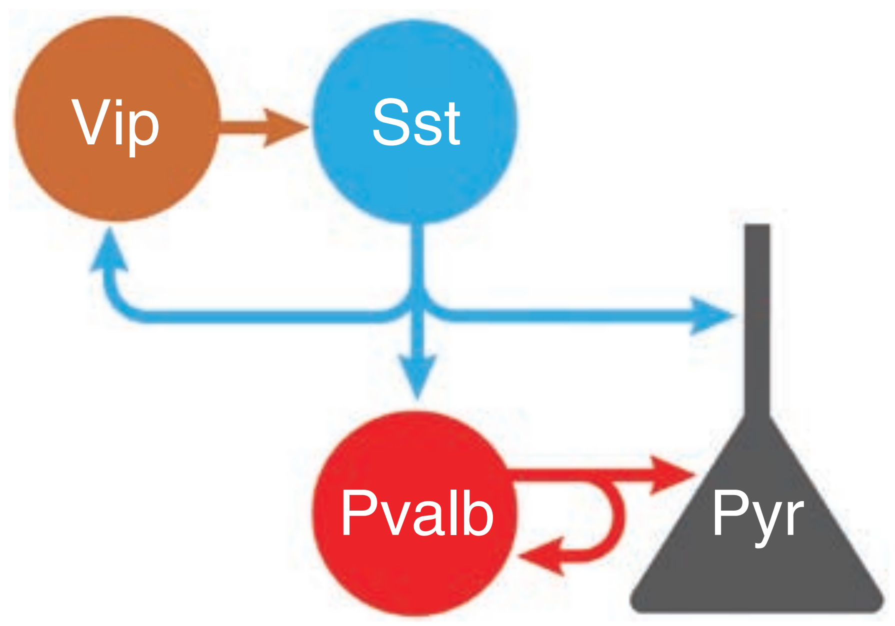
Alexandre Payeur, Jordan Guerguiev, Friedemann Zenke, Blake A. Richards & Richard Naud

Nature Neuroscience **24**, 1010–1019 (2021) | Cite this article

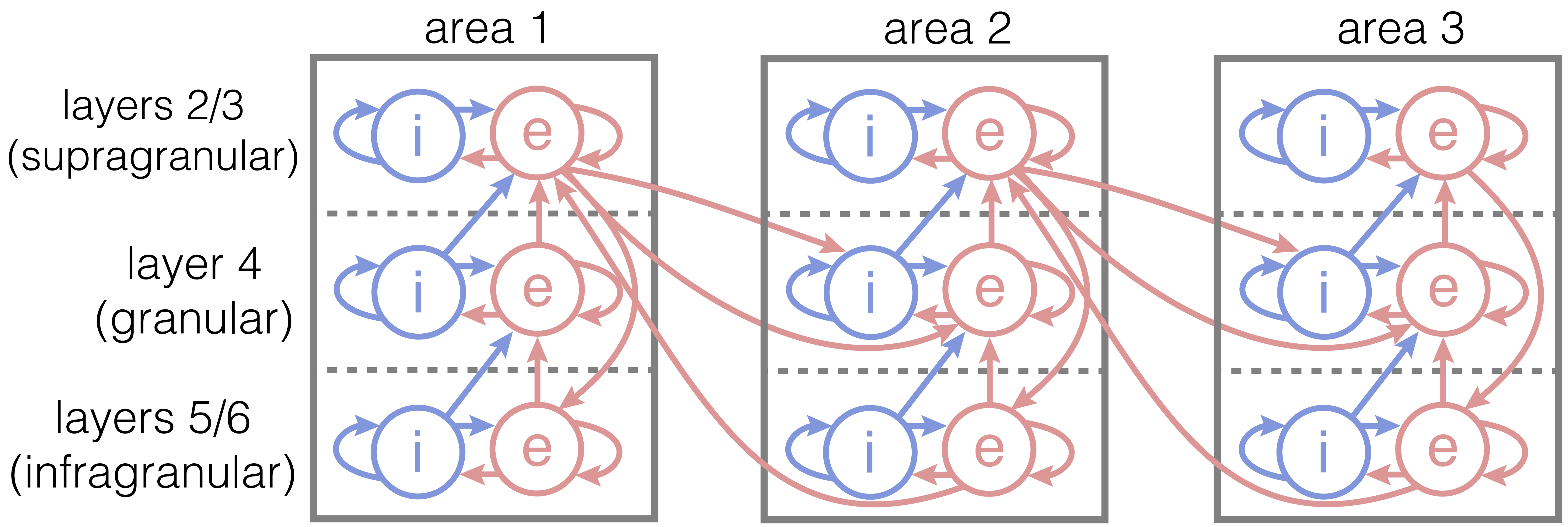
Future plans

- Build a ***framework*** for meta-plasticity in different models with different tasks.



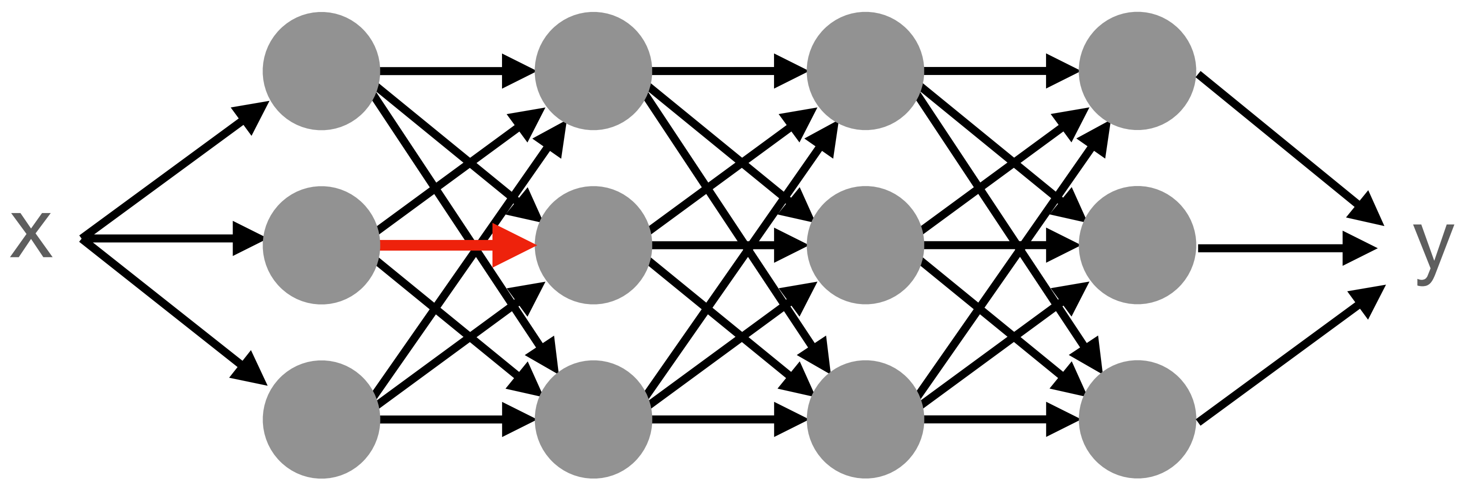


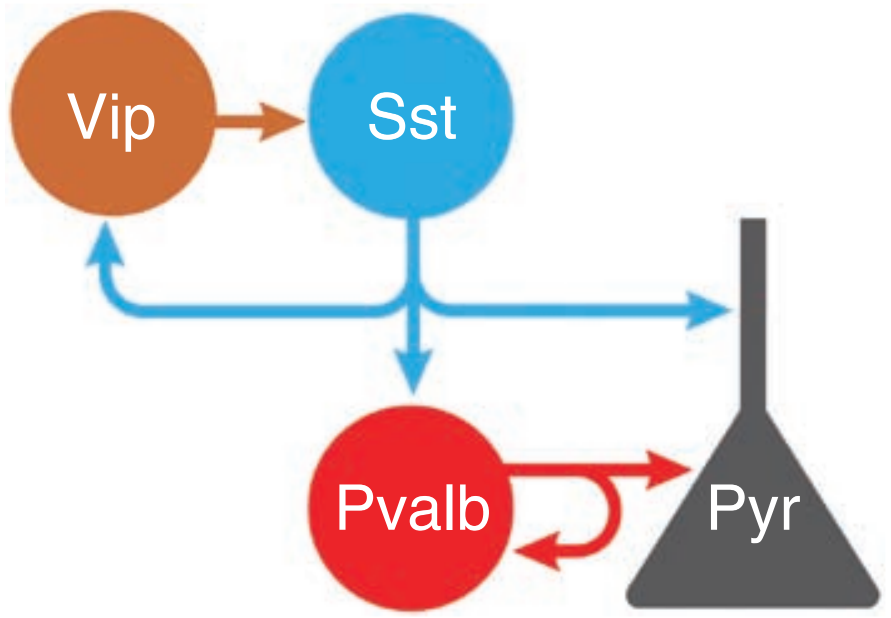
(Pfeffer, et al., 2013)



Future plans

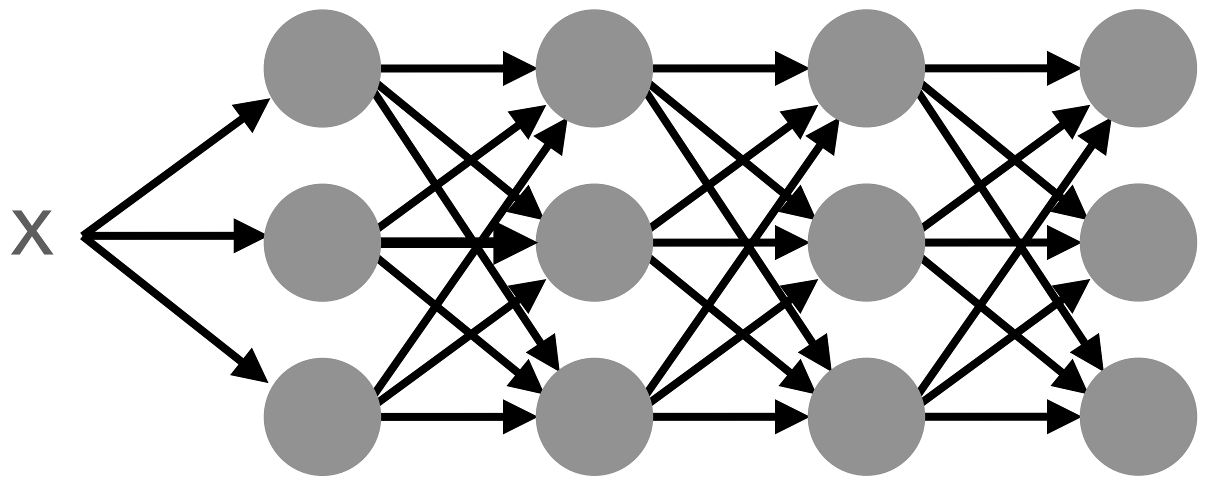
- Build a ***framework*** for meta-plasticity in different models with different tasks.



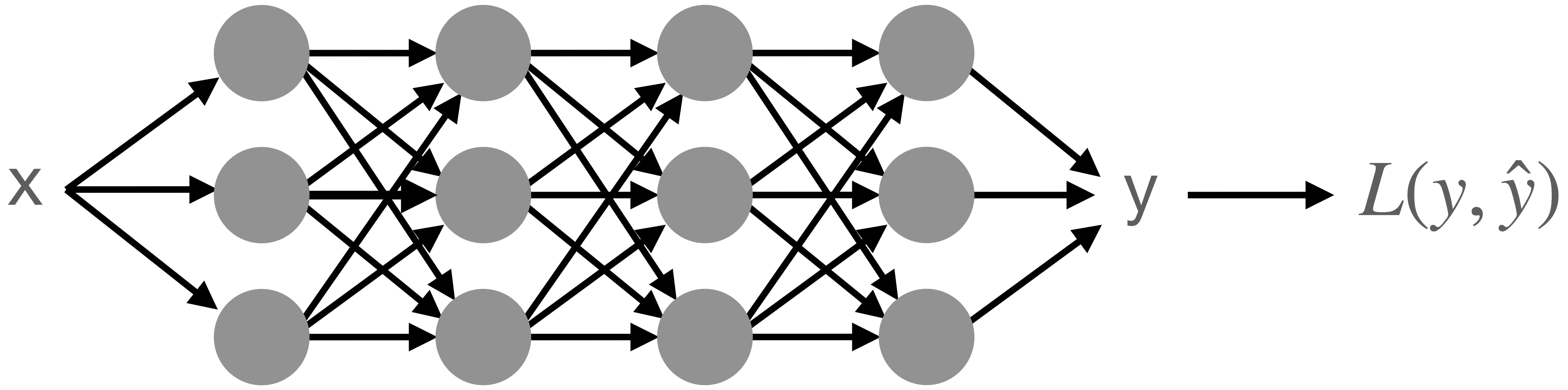


(Pfeffer, et al., 2013)

plasticity trains unsupervised representation



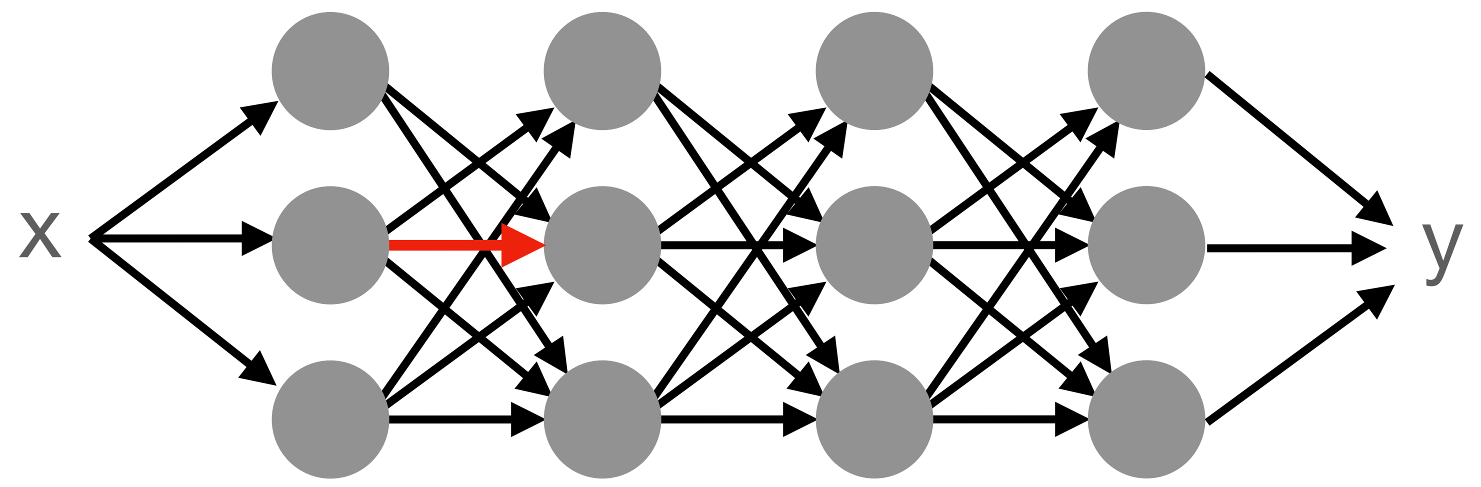
meta-loss evaluates suitability for supervised learning

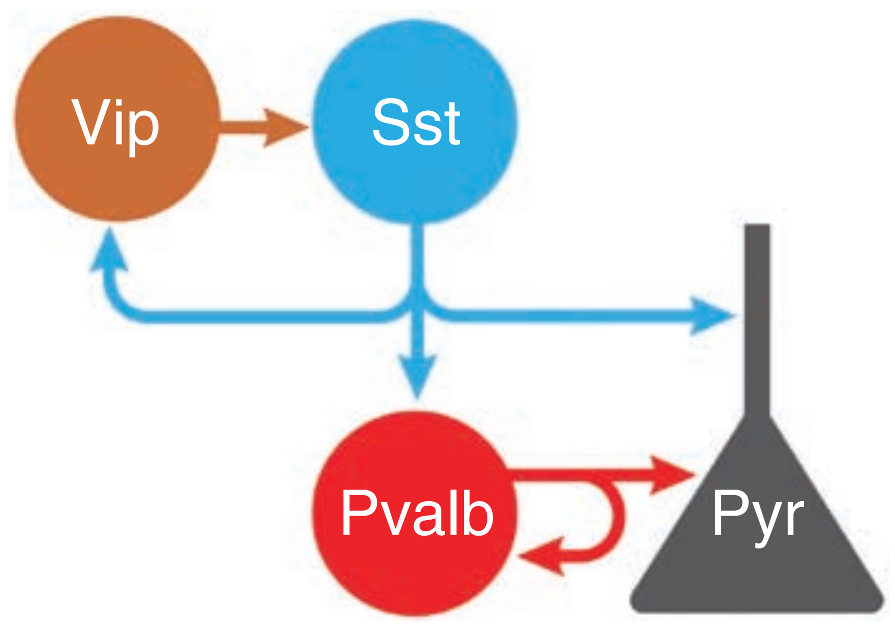


Undergrad researcher: Sarah Duessing

Future plans

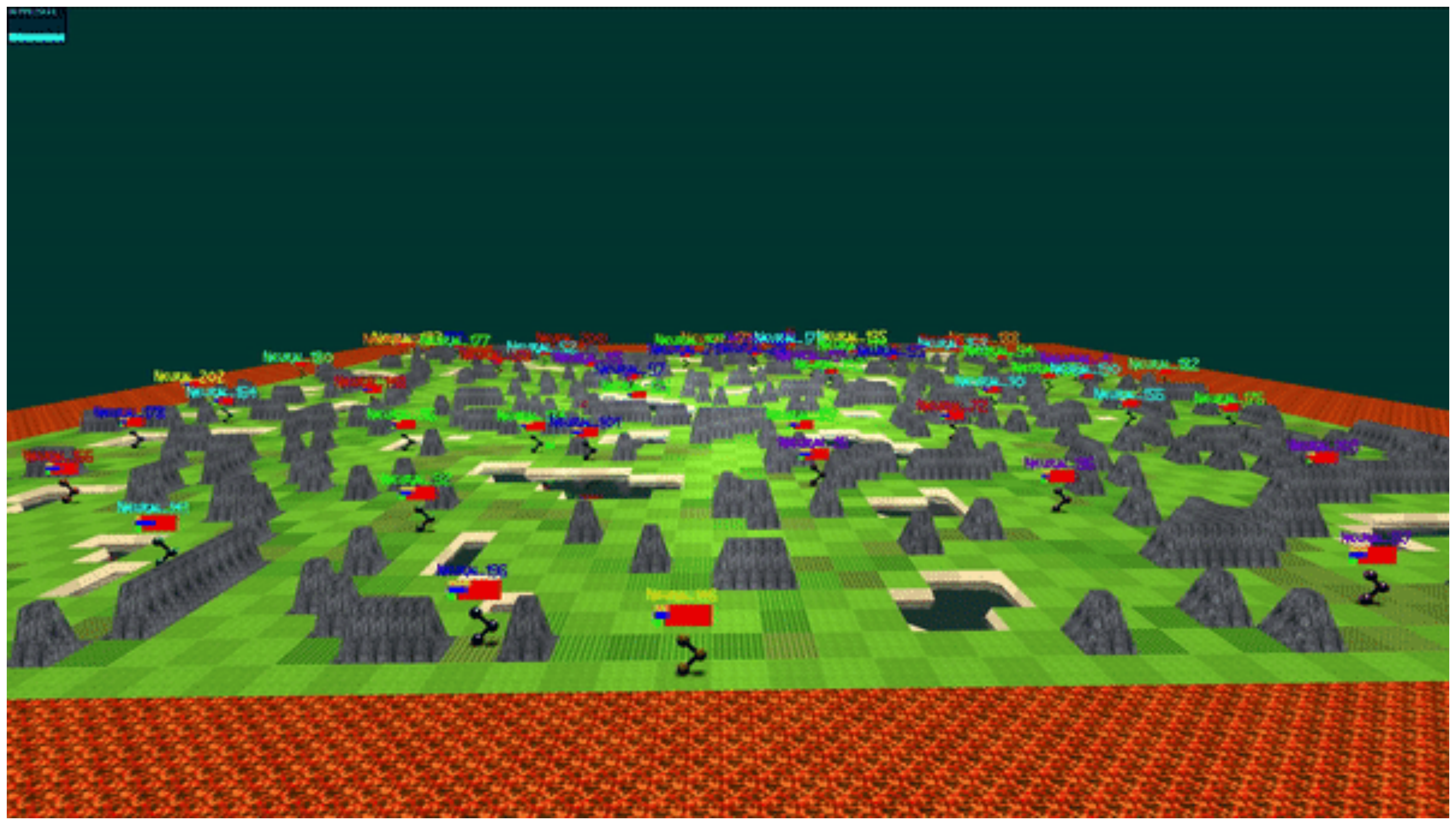
- Build a ***framework*** for meta-plasticity in different models with different tasks.
- ***Evolve*** learning rules in multi-agent environments (gradient-free).





(Pfeffer, et al., 2013)

Neural MMO by OpenAI



Undergrad researcher: Sarah Duessing