DOES FEEDBACK ALIGNMENT WORK AT BIOLOGICAL TIMESCALES?

Anonymous authors
Paper under double-blind review

ABSTRACT

Feedback alignment and related weight-transport–free algorithms of error propagation are often proposed as biologically plausible alternatives to backpropagation, yet they are typically formulated in discrete phases with implicitly synchronized forward and error signals. We develop a continuous-time model of feedback-alignment–type learning in which neural activities and synaptic weights evolve together under coupled first-order dynamics with distinct propagation, plasticity, and decay time constants. Our analysis shows how temporal overlap between presynaptic drive and locally projected error signals governs robustness to timing mismatch. For these two-signal rules, slow weight dynamics cannot bridge feedback delays after stimulus-specific activity has disappeared. We demonstrate learning with millisecond neural dynamics and stimulus presentations lasting tens to hundreds of milliseconds, with simultaneous inference and plasticity.

1 INTRODUCTION

Understanding how biological circuits learn has long been a central challenge at the interface of neuroscience and machine learning. Among the many proposals for biologically plausible learning, a prominent class consists of feedback-alignment–style[1], weight-transport–free algorithms, which includes feedback alignment (FA; Lillicrap et al. (2016)), direct feedback alignment (DFA; Nøkland (2016)), and Kolen–Pollack / weight mirrors (KP; Kolen & Pollack (1994); Akrout et al. (2019)). These algorithms aim to relax the strong requirements of backpropagation, such as exact weight transport, while preserving the ability to optimize deep networks. These algorithms are attractive both as potential models of cortical learning and as approaches for training in neuromorphic or analog hardware.

There are two primary difficulties in connecting FA–type algorithms to biology: learning in biology unfolds continuously in time, and does not rely on separate learning and inference phases. Realistic neurons operate with finite conduction and integration times, so neither inference, error propagation, nor plasticity can be assumed to occur instantaneously or in synchronized steps (Kandel et al., 2000). Unlike digital learning algorithms, biological systems do not alternate between distinct inference and learning phases. By contrast, feedback alignment is cast in discrete steps that alternate between inference and learning phases, effectively assuming that forward and backward signals are globally and instantaneously synchronized (Lillicrap et al., 2016; Nøkland, 2016; Akrout et al., 2019; Scellier & Bengio, 2017). This leaves a basic question unaddressed: *Do FA/KP/DFA–style learning rules still work when implemented as continuous-time processes with biologically relevant neural response timescales?*

In this work, we address this gap directly by constructing a continuous-time model for feedback-alignment–style algorithms. In this model, both neural states and synaptic weights evolve according to coupled first-order differential equations. Each neuron receives a feedforward drive and a modulatory error drive, and updates its synapses through a local two-signal rule. Crucially, inference and learning occur simultaneously: there is no global phase separation, as all state variables evolve under the same ODE. The model is governed by distinct time constants for fast signal propagation, intermediate potentiation, and slow synaptic decay, an ordering that mirrors the hierarchy of timescales in real

[1] In this work, we will use the term "feedback alignment" to refer to the class of feedback-alignment–style algorithms.

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.

neural tissue. Notably, feedback alignment algorithms are inherently non-spiking, and so our continuous-time model also does not attempt to model spikes.

Our model reveals three main insights about feedback alignment in continuous time:

1. **Continuous-time realization.** Feedback-alignment-style learning works in continuous time: inference and learning can proceed simultaneously under coupled differential equations without explicit forward/backward phases.
2. **Overlap as the operative principle.** Learning is governed by temporal overlap between presynaptic drive and locally projected error signals. We derive a closed-form expression for the correctly paired part of the update and show how delays degrade the correct portion of the update.
3. **A limit on delayed credit.** Within this rule family, changing the weight's plasticity rates or decay times cannot recover a stimulus-specific update after its activity has vanished: bridging that gap requires persistent activity or an additional state, such as a synaptic eligibility trace, before the coincidence product is formed.

2 Related Work

Feedback alignment and weight-transport–free learning. The feedback-alignment family was introduced by Lillicrap et al. (2016), who showed that fixed random backward weights are sufficient to support deep learning, and was extended by Nøkland (2016) to direct feedback alignment (DFA), where the output-layer error is broadcast to every hidden layer through fixed random projections. Liao et al. (2016) further dissected the alignment mechanism and showed that the magnitudes of feedback weights largely do not matter, but their signs do, isolating sign concordance rather than weight transport as the operative requirement. Kolen & Pollack (1994) took a complementary route by training the backward weights to converge to the transpose of the forward weights through matched updates and decay, an idea revisited at scale by Akrout et al. (2019) as "weight mirrors," which nearly match backpropagation on ImageNet without weight transport. On the theoretical side, Refinetti et al. (2021) characterized FA learning as a two-phase "align-then-memorise" dynamics in discrete time; our work studies the dynamics of the same family of algorithms when the discrete-time assumption is dropped and signals coevolve under coupled ODEs.

Biological grounding of FA-type rules. The biological plausibility of FA-style algorithms has been explored in several directions. Lillicrap et al. (2020) argued that FA-like mechanisms are consistent with cortical and cerebellar circuitry, and Koplow et al. (2025) showed that such rules align with observed Hebbian and anti-Hebbian plasticity motifs. Ziyin et al. (2025) cast FA, DFA, and KP within a broader heterosynaptic two-signal framework, and Liao et al. (2024) demonstrated that local plasticity rules linking forward and feedback pathways can approximate backpropagation on standard benchmarks. Nakajima et al. (2024) showed that DFA can be realized using physical reservoir dynamics, motivating a careful treatment of the continuous-time and finite-propagation regime in which any analog substrate must operate.

What none of the above analyzes is whether FA-type rules continue to function once the implicit discrete-time, globally synchronized phase structure is removed. Our work addresses that gap directly: we treat the FA family as a continuous-time dynamical system, and identify the temporal-overlap condition that governs whether learning succeeds at biologically realistic timescales.

3 A Continuous-Time Model of Feedback Alignment

The FA–type algorithms feature a forward pathway that performs inference and a backward pathway that transmits error signals. While the original FA algorithm requires a fixed and randomly connected feedback pathway whose topology matches the feedforward, various variants have made the backward pathway much more flexible. For this reason, we study a broad and flexible set of connectivity topologies (see some discussions of this point in Ziyin et al. (2025)). Figure 1 illustrates the continuous-time heterosynaptic neuronal systems we study. The forward pathway, parameterized by $\mathbf{w}$ and with activation σ, maps inputs $\mathbf{x}$ to neuronal output z. The error pathway, parameterized by $\mathbf{v}$, receives an error signal $\boldsymbol{\epsilon}$ and provides a modulatory influence on learning. Learning is heterosynaptic:

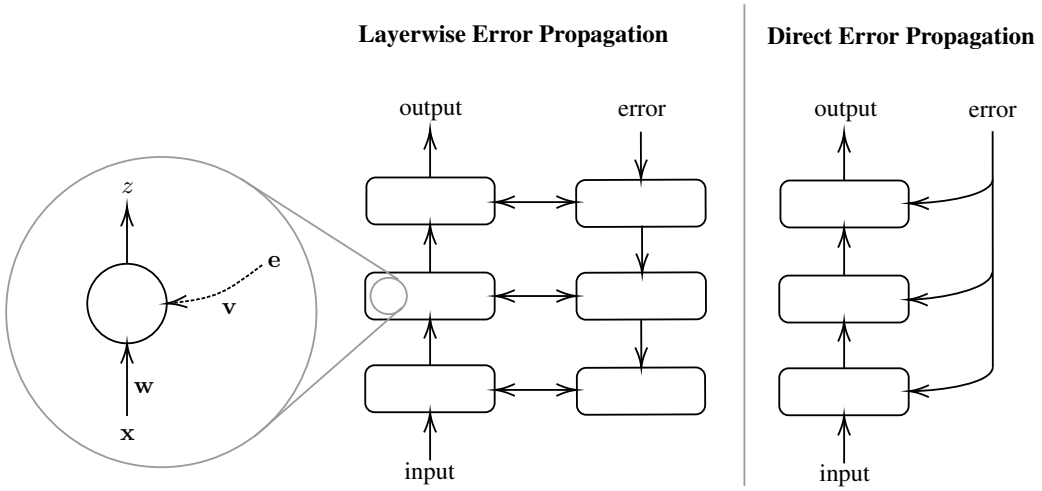


Figure 1: Neuron design with two archetypal topologies. **At left (zoom):** continuous-time heterosynaptic neuron model whose weights evolve according to plasticity rules. Neurons receive forward input $\mathbf{x}$ and produce activated output z with weights $\mathbf{w}$. Error signals $\mathbf{e}$ enter through modulatory weights $\mathbf{v}$ and drive plasticity of $\mathbf{w}$. **In center:** layerwise error propagation topology, where errors are propagated to previous layers via backward weights. **At right:** direct error propagation topology, where error signals are broadcast to previous layers directly.

the forward weights $\mathbf{w}$ update in proportion to the modulatory drive $(\mathbf{v}^\top \boldsymbol{\epsilon})$, while the error weights $\mathbf{v}$ update in proportion to the forward drive $(\mathbf{w}^\top \mathbf{x})$. Let layer l have width d_l. See Appendix A for the per-neuron update rules. Stacking the per-neuron variables into matrices, we can use the following notation: $\mathbf{z}_{l-1}(t) \in \mathbb{R}^{d_{l-1}}$, $\mathbf{z}_l(t) \in \mathbb{R}^{d_l}$, $W_l \in \mathbb{R}^{d_l \times d_{l-1}}$, and error source $\boldsymbol{\epsilon}_l(t)$ to define

$$\dot{\mathbf{z}}_l = \frac{-\mathbf{z}_l + \sigma_l(W_l \mathbf{z}_{l-1})}{\tau_{\text{prop}}}, \quad \dot{W}_l = -\frac{W_l}{\tau^W_{\text{dec}}} + \frac{(V_l \boldsymbol{\epsilon}_l)\mathbf{z}_{l-1}^\top}{\tau^W_{\text{plas}}}, \quad \dot{V}_l = -\frac{V_l}{\tau^V_{\text{dec}}} + \frac{(W_l \mathbf{z}_{l-1})\,\boldsymbol{\epsilon}_l^\top}{\tau^V_{\text{plas}}}. \tag{1}$$

Here τ_{prop} is the neuronal propagation time constant, setting how quickly $\mathbf{z}_l$ relaxes to its driven input. The constants $\tau^W_{\text{plas}}, \tau^V_{\text{plas}}$ control how rapidly synaptic plasticity occurs when presynaptic and modulatory drives coincide. The constants $\tau^W_{\text{dec}}, \tau^V_{\text{dec}}$ govern passive weight decay, setting the forgetting timescale. The center and right panels of Figure 1 show how these neurons connect into multilayer networks. The analytic reduction assumes $\tau_{\text{prop}} \ll T$, so activations approximately settle within a presentation window of length T, and that weight drift within that window is small. Under these conditions outputs approximately track $\mathbf{z}_l \approx \sigma_l(W_l \mathbf{z}_{l-1})$ after their initial transient. Equation 1 displays the idealized two-signal dynamics. The simulations also resolve a filtered local error drive and use a graded gate at the receiving neuron; some runs include additional stabilization terms. The kernel calculation below is exact for the displayed weight equation, while its piecewise-constant overlap reduction approximates the simulated dynamics.

These dynamics can be understood as a continuous-time realization of the heterosynaptic two-signal principle. When neural activity settles within a presentation window and weights change little during that window, integrating the forward-weight equation gives a local outer-product update. The resulting correspondence to FA, DFA, or learned-feedback rules depends on the error routing and on whether V_l is fixed or updated. In our experiments V_l learns; in standard FA and DFA it is fixed. This quasi-static argument does not by itself imply that V_l converges to $W_{l+1}^\top$. Appendix B makes the correspondences explicit.

We simulate training and inference of these models using ODE solvers (Kidger, 2021). Inputs to the network are driven by the dataset's input signals.Denoting the loss gradient as $\mathbf{e} = \partial \mathcal{L}/\partial \mathbf{z}_L$, the error drive in the positive plasticity term has the descent sign $-\mathbf{e}$. Figure 2 shows an example of the output-layer dynamics of early training. Each data point is presented to the network's input neurons for a fixed duration of time (sample time), with interpolation between the images done smoothly over a constant, much shorter duration (buffer time). Rather than impose the corresponding constraints

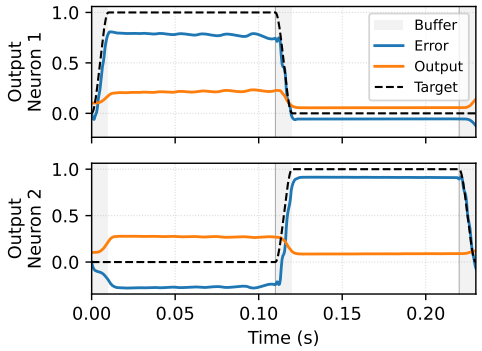


Figure 2: Zoomed-in view of output neuron dynamics during early training. The network input and corresponding error term changes during the *buffer* time periods. Over a single sample, the neuron outputs move only minutely.

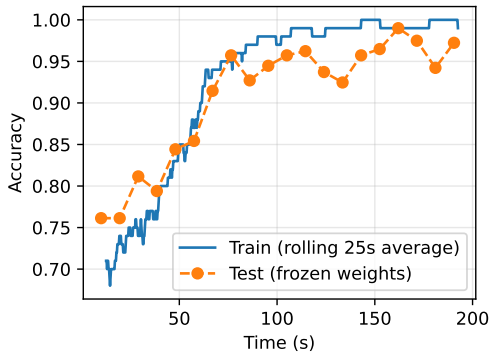


Figure 3: Sample train/test accuracy during training. Training set accuracy is measured as the moving average of instantaneous correct/incorrect predictions on the network input signals. Test set accuracy is measured by evaluating the network with frozen weight dynamics.

and clamps on weights as traditional SGD/KP/FA/DFA prescribe, our experiments leave both W and V free to learn. We find that this makes our network architecture more general without sacrificing task performance. During evaluation, parameters are held fixed and plasticity and decay dynamics are frozen: $\dot{W} = \dot{V} = 0$ and $\boldsymbol{\epsilon} \equiv 0$. Only inference dynamics $\dot{\mathbf{z}}$ are active. The model's prediction is read out at the end of the input presentation window, immediately before the next sample is shown.

4 Continuous-time feedback alignment requires temporal overlap

A central question is: *under what conditions does a synapse receive a correct update?* Two key considerations are (i) the temporal mismatch between input and error signals, and (ii) the rate at which the input changes. Figure 4 illustrates the dynamics of a final-layer weight when the error signal arrives earlier or later than the corresponding input. During the mismatch period, the instantaneous weight update is incorrect, leading to a cumulative update that deviates from the non-delayed case.

To isolate the timing mechanism, write $u_l(t) = \mathbf{z}_{l-1}(t)$ for the presynaptic drive and $m_l(t) = V_l(t)\boldsymbol{\epsilon}_l(t)$ for the local error drive in the idealized equation 1. Variation of constants over a window of length T gives the plasticity-driven change

$$W_l(T) - e^{-T/\tau^W_{\text{dec}}} W_l(0) = \frac{1}{\tau^W_{\text{plas}}} \int_0^T e^{-(T-t)/\tau^W_{\text{dec}}} m_l(t) u_l(t)^\top \, dt. \tag{2}$$

The kernel carries the weight-decay time constant; $1/\tau^W_{\text{plas}}$ is an update-rate factor outside the integral. This kernel retains weight changes after they occur, but does not store either drive before their product is formed. At synapse $(i \to j)$, the integrand is $m_{l,j}(t)u_{l,i}(t)$.

Matched and mismatched pairings. For the analytic approximation, assume an i.i.d. stream of samples, quasi-static weights within each presentation window, and piecewise-constant drives once fast propagation transients have settled. Let u_n and m_n be the drives associated with sample n. An externally imposed offset $\Delta > 0$ delays its error drive relative to its input; $\Delta < 0$ advances it. For $|\Delta| < T$, its own error occupies $C(\Delta) = [0, T] \cap [\Delta, \Delta + T]$, while the complementary interval $I(\Delta) = [0, T] \setminus C(\Delta)$ contains a neighboring sample's error. Define $U_{\text{match}} = \mathbb{E}[m_n u_n^\top]$ and $U_{\text{mis}}(\Delta) = \mathbb{E}[m_{n'} u_n^\top]$, where n' is the neighboring sample selected by the sign of Δ. The expectation is over the sample stream; neither term vanishes merely because it is averaged. With

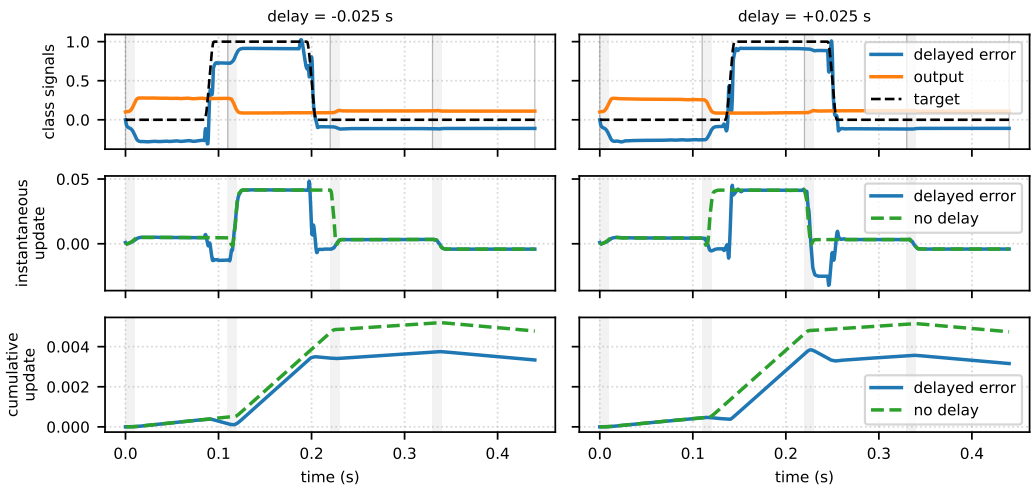


Figure 4: Single-neuron dynamics with different relative timings of the error and input signals. **Left:** error signal arrives early. **Right:** error signal is delayed. The bottom panel of both plots depicts the cumulative weight change and shows that in the presence of delay, the weight accumulates a biased gradient update compared to the case where there was no error delay.

$k(t) = e^{-(T-t)/\tau_{\text{dec}}^W}$, the idealized update for $|\Delta| < T$ is

$$\mathbb{E}[\Delta W_l^{\text{drive}}] = \frac{K_C(\Delta) U_{\text{match}} + K_I(\Delta) U_{\text{mis}}(\Delta)}{\tau_{\text{plas}}^W}, \quad K_C = \int_{C(\Delta)} k(t)\,dt, \quad K_I = \int_{I(\Delta)} k(t)\,dt. \tag{3}$$

Here $\Delta W_l^{\text{drive}}$ denotes the left-hand side of equation 2. Finite propagation smooths the drives, so the decomposition into two constant matrices is an approximation to the simulated dynamics.

Flat-kernel overlap law. For $|\Delta| < T$, put $t_0 = \max(0, \Delta)$ and $t_1 = \min(T, \Delta + T)$. The weight on the matched term is

$$K_C(\Delta) = \tau_{\text{dec}}^W \left(e^{-(T-t_1)/\tau_{\text{dec}}^W} - e^{-(T-t_0)/\tau_{\text{dec}}^W} \right); \qquad K_C(\Delta) = 0 \text{ if } |\Delta| \geq T. \tag{4}$$

Since $K_C + K_I = K_T := \int_0^T k(t)\,dt$, the matched fraction is $g(\Delta) = K_C/K_T$. In the delay–duration sweeps $T \leq 0.25\,\text{s}$ and $\tau_{\text{dec}}^W = 10^3\,\text{s}$, so the kernel is nearly flat and

$$g(\Delta) \simeq \left(1 - \frac{|\Delta|}{T}\right)_+. \tag{5}$$

This triangle describes the *fraction of the update window paired with the correct sample*, not the sign or magnitude of the total update. For $|\Delta| \geq T$ the matched term is absent, while mismatched pairings may involve more than one neighboring sample. Decreasing overlap increases the weight assigned to the mismatched contribution, which a global learning-rate rescaling cannot remove. Finite propagation and the full training dynamics need not place an accuracy threshold exactly at $|\Delta| = T$.

In an isolated trial where stimulus-specific activity has ended before delayed feedback begins, $m_l(t)u_l(t)^\top$ contains no informative product at any time. Changing τ_{plas}^W or τ_{dec}^W cannot recover that missing association. In a continuous sample stream, the input drive exists but belongs to another sample, yielding the mismatch term above instead of a zero total update.

5 Feedback Alignment Works at Biological Timescales

Running these experiments is computationally demanding. Each training and evaluation run requires numerical integration of a high-dimensional, stiff system of coupled ODEs. Neuronal activations evolve on millisecond timescales, while meaningful learning unfolds over hundreds to thousands of

seconds. This range of time scales spans six orders of magnitude, forcing solvers to take very small steps across very long horizons.

We implement our simulations in JAX (Bradbury et al., 2018) and use the `Diffrax` library (Kidger, 2021) for ODE integration. Although we do not rely on JAX's autodifferentiation, its just-in-time compilation and efficient array operations provide substantial performance gains for these simulations. We use adaptive PID step control; the MNIST timescale sweep uses the implicit–explicit KenCarp4 solver, while the delay–duration sweeps use the explicit Tsit5 solver (Tsitouras, 2011). Appendix C gives the implementation details.

Some of the parameter sweeps reported here, particularly the heatmaps over error delay and sample duration, require over 500 compute hours to complete. This reflects the intrinsic difficulty of simulating continuous-time learning rules, where propagation, plasticity, and decay processes must be faithfully resolved simultaneously. These results represent a substantial computational effort and demonstrate that continuous-time neural networks can be simulated effectively despite these challenges.

In this section, we evaluate three experimental regimes. Section 5.1 demonstrates learning when error signals overlap with inputs. Section 5.2 demonstrates that deeper networks accumulate propagation lag, making them less tolerant to error delays and requiring longer sample times for stable learning. Finally, Section 5.3 examines how propagation and plasticity rates affect learning at a fixed stimulus duration.

5.1 Direct Error Routing

The direct error routing topology sets the local error at each layer to be the global error (from the output layer), $\boldsymbol{\epsilon}_l = \mathbf{e}_L$. This uses the routing of Direct Feedback Alignment, with the backward weights V_l also learned. Figure 5 examines the accuracy of the direct error routing topology on the 7×7 downsampled MNIST dataset as the error-signal delay and sample duration are varied.

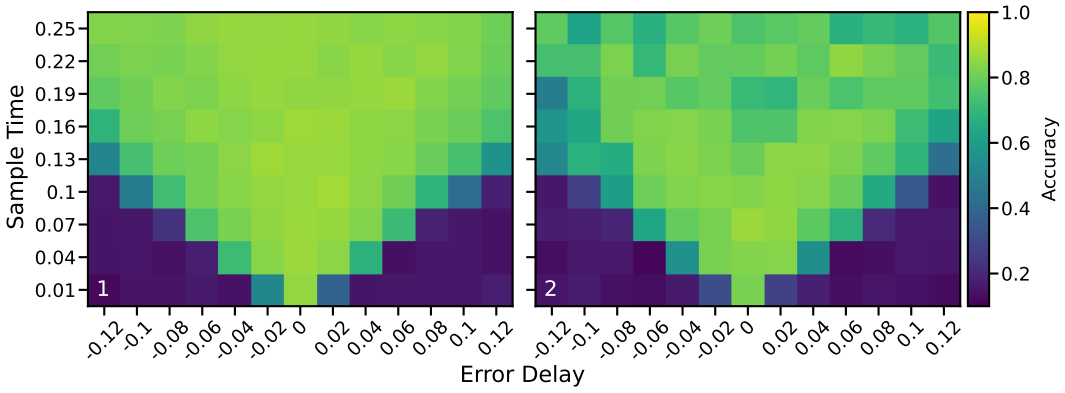


Figure 5: Evaluation of the direct error routing topology on the 7×7 downsampled MNIST dataset. The x-axis denotes the temporal delay between input signal and label: negative values indicate the label arrives before the input, while positive values indicate the label arrives after the input. At left is a network with 1 hidden layer of 49 neurons, and at right is a network with 2 hidden layers of 49 and 32 neurons. The learning is fairly robust until the delay exceeds the sample time.

In the direct error routing topology, each layer receives a direct copy of the error signal. As a result, the feedback path from the output to each hidden layer contains one synaptic projection. Learning fails when the error signal delay is approximately equal to or longer than the sample duration, since the error signal has zero overlap with its corresponding input. In these regions (accuracy $\sim 10\%$), the idealized overlap model leaves only mismatched input-label pairings in the plasticity drive. Transitional regions ($\sim 70\%$ accuracy) arise when error and input partially overlap: some useful updates occur, but they are mixed with incorrect updates from mismatched periods. These networks have fixed $\tau_{\text{plas}} = 10\,\text{s}$ and $\tau_{\text{dec}} = 1000\,\text{s}$ while the sample time T lies in 0.01–0.25 s, putting them in the flat-kernel limit $T \ll \tau_{\text{dec}}$. As a result, learning responds fairly symmetrically to both early and late error signals.

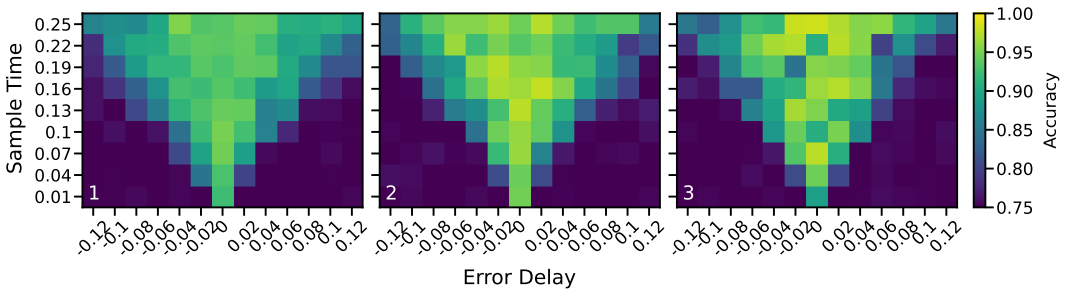


Figure 6: Evaluation of the layerwise error routing topology on the circle dataset as error signal delay and sample duration are swept. The network topologies increase in the number of hidden layers, where each added hidden layer has 24 neurons. From left to right, the networks have one hidden layer of 24 ReLU neurons, two hidden layers of 24 ReLU neurons, and three hidden layers of 24 ReLU neurons. Layerwise error routing imposes stricter requirements on delay due to longer error propagation paths.

5.2 Layerwise Error Routing

The layerwise error routing topology propagates error from the output layer step by step to previous layers via $V_l \epsilon_l$. This is a KP- or weight-mirror-style scheme where the backward weights V_l are learned. Figure 6 examines the accuracy of the layerwise error routing topology on a synthetic two-dimensional circles dataset. This task is more nonlinear than the 7×7 downsampled MNIST task, and requires deeper ReLU networks to represent an accurate decision boundary.

Unlike the direct error routing topology, the layerwise error routing topology suffers more from finite propagation times: not only does the input signal have to propagate to the output layer, but the error signal must also propagate backward from the output layer back to the input layer. The heatmaps show that these deeper networks require longer sample times and are more sensitive to delay in the error signal. Finite propagation speed means that deeper networks accumulate lag, degrading the learning signal especially in early layers. Longer sample durations mitigate this effect by increasing the window where inputs and their true errors coincide. In the language of Section 4, this increases T relative to a fixed Δ, increasing the matched fraction $g(\Delta)$. In the lower triangular regions of the heatmap (large absolute delays), overlap is insufficient and learning fails entirely.

5.3 Sensitivity to Propagation and Plasticity Rates

A central motivation for these algorithms is their relevance to biological learning. Fast neural responses reflect synaptic receptor kinetics and membrane filtering (Destexhe et al., 1998). At hippocampal principal-neuron–interneuron synapses, AMPA-mediated conductances rise and decay on submillisecond timescales, producing excitatory postsynaptic potentials with a half-duration of approximately 4 ms (Geiger et al., 1997). In piriform cortex, inhibitory postsynaptic currents have fast and slow decay components around 10 and 40 ms, respectively (Kapur et al., 1997). Synaptic strength is also regulated through AMPA receptor trafficking and phosphatase-dependent recycling (Ehlers, 2000), activity-dependent receptor accumulation (O'Brien et al., 1998), and postsynaptic protein turnover (Ehlers, 2003). These processes are distinct from fast synaptic signaling; homeostatic changes in synaptic strength have, for example, been measured over a 48 h period (Turrigiano et al., 1998). This separation between fast signaling and slower synaptic regulation motivates studying neural activity and weight adaptation on distinct timescales.

Figure 7 evaluates two layerwise error routing networks at a stimulus duration of $T = 50$ ms, displaying propagation constants τ_{prop} from 0.5 to 30 ms and plasticity parameters τ_{plas} from 1.45 to 10 s, with $\tau_{\text{dec}} = 1000\,\text{s} \approx 17$ min held fixed. In our model, τ_{prop} sets the relaxation time of neural activity, τ_{plas} controls the strength of the plasticity drive, and τ_{dec} sets the timescale of passive weight decay. Increasing τ_{plas} therefore slows weight adaptation for a given pair of drives. We find that τ_{dec} plays little role in the learning dynamics once it is sufficiently larger than τ_{prop} and τ_{plas}; we therefore hold it fixed here. Recorded accuracy is generally higher for faster propagation and slower weight adaptation, with the deeper network showing greater sensitivity to these parameters. These

results demonstrate feedback-alignment-style learning with activity-relaxation times that overlap measured fast neural response timescales.

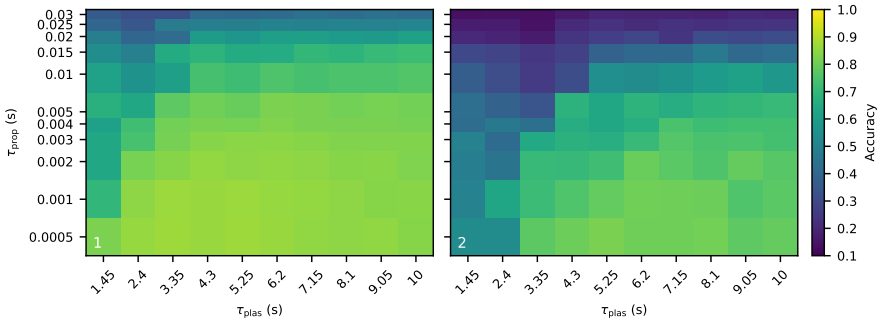


Figure 7: Evaluation of a layerwise error routing network on the 7×7 MNIST dataset. **Left:** 1 hidden layer (49 neurons). **Right:** 2 hidden layers (49 and 32 neurons). Propagation and plasticity time constants are varied at a fixed presentation duration of $T = 50\,\text{ms}$ and decay time constant of $\tau_{\text{dec}} = 1000\,\text{s}$. Colours show peak test accuracy over available evaluation checkpoints, averaged across runs, including those that terminated before the training horizon.

6 Discussion

We asked whether feedback-alignment learning can operate in continuous time, without artificial phase separation and at realistic neural timescales. To answer this question, we formulated a coupled ODE model of feedback alignment and characterized its behavior as a function of error delays and plasticity time constants. By embedding neural activity and synaptic plasticity within a single set of differential equations, we show that learning can emerge from continuous-time dynamics without any explicit synchronization.

Our model identifies temporal pairing as an additional constraint on feedback-driven learning. In a continuous sample stream, shrinking the temporal overlap between an input and its corresponding error increases the interval in which a synapse pairs one sample's presynaptic drive with an error computed from another sample's label; once correct overlap disappears, the matched component vanishes and only mismatched updates remain. In an isolated trial with no stimulus-specific activity when feedback arrives, the informative product is zero.

This perspective clarifies the functional role of timescale separation. Fast propagation relative to the stimulus duration lets neural states track inputs, slow weight adaptation permits a quasi-static approximation within each presentation window, and $\tau_{\text{dec}} \gg T$ makes decay negligible within that window. Figure 7 examines the sensitivity of learning to propagation and plasticity rates in the coupled dynamics.

We find that feedback alignment-style learning is possible without globally separated forward and backward phases. Scaling to deeper networks remains unresolved: we find that training becomes unstable in deeper networks, and previous feedback-alignment rules have required stabilization in discrete-time work (Akrout et al., 2019; Moskovitz et al., 2018; Webster et al., 2020). Common stabilization methods such as batch normalization and layer normalization rely on minibatch statistics or nonlocal computations across neurons and do not transfer directly to a local, continuous-time model. Developing effective stabilization methods for these models remains an important open problem for future investigation. The experiments here are evidence for the timing mechanism at the studied depths, not a demonstration of scalable training.

These results could extend beyond biological modeling. Any analog physical substrate must contend with finite propagation timescales and heterogeneous device dynamics. Our analysis suggests that such systems need not enforce exact symmetry between forward and backward pathways. Instead, they must preserve sufficient temporal overlap between signals driving synaptic change. This overlap condition is therefore an important constraint on continuous-time learning in physical systems. More broadly, this work connects error-propagation algorithms to the theory of dynamical

systems. Learning can be understood as the accumulation of temporally filtered correlations under coupled state–parameter evolution. In this formulation, sample-matched updates depend on sustained coincidence between presynaptic and error drives. This suggests a unified lens through which to interpret cortical learning, feedback-alignment algorithms, and hardware implementations as instances of the same dynamical principle.

AI Use Statement

Generative AI tools were used to assist with writing, experiments, and theory. All AI-assisted writing, experimental work, and theoretical content was checked by the authors. The authors take responsibility for the final content of this work.

Reproducibility Statement

The continuous-time model is specified in Section 3, with per-neuron dynamics and discrete-time limits in Appendices A and B. Appendix C describes the datasets, network architectures, initialization, evaluation procedure, and numerical solver settings. Appendix D specifies the discrete-time baselines and their comparison with the continuous-time models.

References

Mohamed Akrout, Collin Wilson, Peter Humphreys, Timothy Lillicrap, and Douglas B Tweed. Deep learning without weight transport. *Advances in neural information processing systems*, 32, 2019.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/jax-ml/jax`.

Alain Destexhe, Zachary F Mainen, Terrence J Sejnowski, et al. Kinetic models of synaptic transmission. *Methods in neuronal modeling*, 2:1–25, 1998.

Michael D Ehlers. Reinsertion or degradation of ampa receptors determined by activity-dependent endocytic sorting. *Neuron*, 28(2):511–525, 2000.

Michael D Ehlers. Activity level controls postsynaptic composition and signaling via the ubiquitin-proteasome system. *Nature neuroscience*, 6(3):231–242, 2003.

Jörg RP Geiger, Joachim Lübke, Arnd Roth, Michael Frotscher, and Peter Jonas. Submillisecond ampa receptor-mediated signaling at a principal neuron–interneuron synapse. *Neuron*, 18(6): 1009–1023, 1997.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings, 2010.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 315–323. JMLR Workshop and Conference Proceedings, 2011.

Eric R Kandel, James H Schwartz, Thomas M Jessell, Steven Siegelbaum, A James Hudspeth, Sarah Mack, et al. *Principles of neural science*, volume 4. McGraw-hill New York, 2000.

A Kapur, RA Pearce, WW Lytton, and LB Haberly. Gabaa-mediated ipscs in piriform cortex have fast and slow components with different properties and locations on pyramidal cells. *Journal of neurophysiology*, 78(5):2531–2545, 1997.

Patrick Kidger. *On Neural Differential Equations*. PhD thesis, University of Oxford, 2021.

John F Kolen and Jordan B Pollack. Backpropagation without weight transport. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, volume 3, pp. 1375–1380. IEEE, 1994.

David Koplow, Tomaso Poggio, and Liu Ziyin. Emergence of hebbian dynamics in regularized non-local learners. *arXiv preprint arXiv:2505.18069*, 2025.

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

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 2002.

Qianli Liao, Joel Z Leibo, and Tomaso Poggio. How important is weight symmetry in backpropagation? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Qianli Liao, Liu Ziyin, Yulu Gan, Brian Cheung, Mark Harnett, and Tomaso Poggio. Self-assembly of a biologically plausible learning circuit. *arXiv preprint arXiv:2412.20018*, 2024.

Timothy P Lillicrap, Daniel Cownden, Douglas B Tweed, and Colin J Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature communications*, 7(1): 13276, 2016.

Timothy P Lillicrap, Adam Santoro, Luke Marris, Colin J Akerman, and Geoffrey Hinton. Backpropagation and the brain. *Nature Reviews Neuroscience*, 21(6):335–346, 2020.

Theodore H Moskovitz, Ashok Litwin-Kumar, and L F Abbott. Feedback alignment in deep convolutional networks. *arXiv preprint arXiv:1812.06488*, 2018.

Mitsumasa Nakajima, Katsuma Inoue, Kenji Tanaka, Yasuo Kuniyoshi, Toshikazu Hashimoto, and Kohei Nakajima. Reservoir direct feedback alignment: deep learning by physical dynamics. *Communications Physics*, 7(1), 2024.

Arild Nøkland. Direct feedback alignment provides learning in deep neural networks. *Advances in neural information processing systems*, 29, 2016.

Richard J O'Brien, Sunjeev Kamboj, Michael D Ehlers, Kenneth R Rosen, Gerald D Fischbach, and Richard L Huganir. Activity-dependent modulation of synaptic ampa receptor accumulation. *Neuron*, 21(5):1067–1078, 1998.

Maria Refinetti, Stephane d'Ascoli, Ruben Ohana, and Sebastian Goldt. Align, then memorise: the dynamics of learning with feedback alignment. In *International Conference on Machine Learning*, pp. 8925–8935. PMLR, 2021.

Benjamin Scellier and Yoshua Bengio. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in computational neuroscience*, 11:24, 2017.

Ch Tsitouras. Runge–kutta pairs of order 5 (4) satisfying only the first column simplifying assumption. *Computers & mathematics with applications*, 62(2):770–775, 2011.

Gina G Turrigiano, Kenneth R Leslie, Niraj S Desai, Lana C Rutherford, and Sacha B Nelson. Activity-dependent scaling of quantal amplitude in neocortical neurons. *Nature*, 391(6670): 892–896, 1998.

Matthew Bailey Webster, Jonghyun Choi, et al. Learning the connections in direct feedback alignment. *arXiv preprint*, 2020.

Liu Ziyin, Isaac Chuang, and Tomaso Poggio. Heterosynaptic circuits are universal gradient machines. *arXiv preprint arXiv:2505.02248*, 2025.

A Dynamics of a Single Neuron

The single-neuron version of the dynamics presented in equation 1:

$$\dot{z} = \frac{-z + \sigma(\mathbf{w}^\top \mathbf{x})}{\tau_{\text{prop}}}, \quad \dot{\mathbf{w}} = -\frac{\mathbf{w}}{\tau^W_{\text{dec}}} + \frac{(\mathbf{v}^\top \boldsymbol{\epsilon})\,\mathbf{x}}{\tau^W_{\text{plas}}}, \quad \dot{\mathbf{v}} = -\frac{\mathbf{v}}{\tau^V_{\text{dec}}} + \frac{(\mathbf{w}^\top \mathbf{x})\,\boldsymbol{\epsilon}}{\tau^V_{\text{plas}}}.$$

B Relation to discrete-time updates

Equation 1 specifies a two-signal continuous-time model; its quasi-static forward-weight step has the outer-product form of several discrete algorithms after choices about feedback routing, gating, and whether V_l is fixed or learned. These choices are part of the correspondence, not consequences of taking a limit. Let $\mathbf{a}_l = W_l \mathbf{z}_{l-1}$ and let $\mathbf{r}_L = -\partial \mathcal{L}/\partial \mathbf{a}_L$ be the output-layer *descent* residual, consistent with the positive plasticity term in equation 1. For a hidden layer, write the local descent drive as

$$\mathbf{r}_l = q_l(\mathbf{a}_l) \odot (V_l \mathbf{s}_l), \qquad \Delta W_l \propto \mathbf{r}_l \mathbf{z}_{l-1}^\top, \tag{6}$$

where $\mathbf{s}_l$ is the routed error source and the gate q_l is computed at the receiving neuron. Thus neuron j forms $q_l(a_{l,j}) \sum_k (V_l)_{jk} s_{l,k}$ locally; a gate vector need not be transported through the feedback pathway. The simulations use a graded surrogate $q_l(a) = \alpha + (1-\alpha)[1 + \tanh(a/k_{\text{gate}})]/2$ with a small leak α, not a binary ReLU mask or an exact derivative of the smoothed activation. The exact discrete correspondences below use $q_l = \sigma_l'$ instead.

Backpropagation. Set $V_l = W_{l+1}^\top$, route $\mathbf{s}_l = \mathbf{r}_{l+1}$, and use $q_l = \sigma_l'$. Then $\mathbf{r}_l$ is the negative pre-activation loss gradient and the positive outer-product step in equation 6 is gradient descent.

Feedback alignment (FA). Hold each V_l fixed at a random matrix of shape $d_l \times d_{l+1}$, route $\mathbf{s}_l = \mathbf{r}_{l+1}$, and use $q_l = \sigma_l'$. This gives the standard layerwise FA forward-weight update (Lillicrap et al., 2016). Fixed feedback means omitting the $\dot{V}_l$ plasticity equation.

Direct feedback alignment (DFA). Hold each V_l fixed at a random matrix of shape $d_l \times d_L$, broadcast $\mathbf{s}_l = \mathbf{r}_L$, and use $q_l = \sigma_l'$. This gives the standard DFA forward-weight update (Nøkland, 2016); again, V_l does not learn.

Learned-feedback variants. In our experiments both W_l and V_l evolve according to local two-signal drives, as in weight-mirror or KP-style proposals (Kolen & Pollack, 1994; Akrout et al., 2019; Ziyin et al., 2025). This shares their learned-feedback motivation but is not identical to fixed-feedback FA or DFA. Nor does the quasi-static argument alone establish that V_l converges to $W_{l+1}^\top$; that would require an additional analysis of the coupled updates and stabilization.

C Datasets & Methodology

We use two datasets in our evaluation of these models. The 7×7 downsampled MNIST dataset is the standard MNIST (LeCun, 1998) dataset that has been downsampled with 4×4 average pooling. The circles dataset is generated as two concentric radial regions with alternating labels and uniform sampling over disk area; the inner region occupies 25% of that area. It is well established that simple linear classifiers achieve surprisingly high accuracy on MNIST, with only modest improvements from deeper architectures (LeCun et al., 2002). We find a linear softmax regressor on 7×7 downsampled MNIST achieves over 89% test accuracy. In contrast, a logistic regressor on the 2-circle concentric-rings dataset achieves only 75% test accuracy, which corresponds to the class imbalance itself (75% outer ring, 25% inner ring).

Network architectures. On 7×7 MNIST (49-dim input, 10 classes), the 1-layer network has hidden widths $[49]$ and the 2-layer network has hidden widths $[49, 32]$; this configuration is used for the direct error routing sweep (Figure 5), the propagation–plasticity sweep (Figure 7), and Table 1. On the 2D circles task (2-dim input, 2 classes), the 1-, 2-, and 3-layer networks have hidden widths $[24]$, $[24, 24]$, and $[24, 24, 24]$ respectively; this configuration is used for the layerwise error routing sweep (Figure 6), Table 2, and the delay-dithering comparison (Figure 8). Hidden units use a smoothed leaky-ReLU activation and a graded local error gate; the output layer applies softmax for cross-entropy loss.

W are initialized with rowwise He scaling for nonlinear units and Xavier scaling for linear rows (Glorot & Bengio, 2010). V are initialized to positive, fan-in-scaled values on active feedback connections. Classification decisions are read out from output neurons at the very end of each input sample's presentation. Evaluation is done on frozen W and V dynamics. Each heatmap data point is the average of 3-5 runs, depending on the experiment.

We integrate the continuous-time dynamics using Diffrax (Kidger, 2021) with a PID adaptive step-size controller ($\texttt{rtol} = 2 \times 10^{-3}$, $\texttt{atol} = 10^{-5}$ in the cited sweep runners). The MNIST timescale sweep uses KenCarp4, treating the linear relaxation and decay terms implicitly; the delay–duration sweeps use Tsit5, an explicit fifth-order Runge–Kutta method (Tsitouras, 2011). Adaptive step sizing resolves rapid transients around input switches and error onsets while allowing larger steps during slowly varying segments.

D Comparison to SGD

We present the following comparisons not to argue that our continuous-time networks outperform vanilla SGD, but to verify that they reach comparable levels of performance under matched training conditions, a basic sanity check that the heterosynaptic learning scheme is actually working as a credible online learning rule.

To make the comparison meaningful, we train baseline multi-layer perceptrons (MLPs) with the same hidden unit counts as our continuous-time networks, and deliberately match the discrete- and continuous-time training conditions as closely as possible. Each baseline sees exactly the same training samples in the same order as the corresponding continuous-time network, with one online update per sample (batch size 1) and a single pass through the data. Baselines use vanilla stochastic gradient descent on batch size 1 (no momentum or other optimizers); the per-cell learning rate is the value selected from a sweep over $\{10^{-4}, 3{\cdot}10^{-4}, 10^{-3}, 3{\cdot}10^{-3}, 10^{-2}, 3{\cdot}10^{-2}, 10^{-1}, 3{\cdot}10^{-1}\}$ that maximized test accuracy, giving the baseline its strongest plausible result. The baselines use ReLU activations (Glorot et al., 2011), while the continuous-time models use smoothed leaky-ReLU activations.

Table 1 reports accuracies on the 7×7 MNIST dataset. The continuous-time and discrete-time test accuracies are within one percentage point of one another at both depths. On a task this close to linearly separable, this near-tie is the expected outcome: a well-tuned online SGD baseline already nearly saturates what one-pass training on 7×7 MNIST can deliver, and the dataset yields diminishing returns from additional layers.

	Train Accuracy		**Test Accuracy**	
Model	Discrete-time	Continuous-time	Discrete-time	Continuous-time
1-layer	0.9223 ± 0.0019	$\mathbf{0.9318 \pm 0.0028}$	0.9126 ± 0.0042	$\mathbf{0.9209 \pm 0.0011}$
2-layer	0.9105 ± 0.0294	$\mathbf{0.9318 \pm 0.0032}$	$\mathbf{0.8990 \pm 0.0308}$	0.8919 ± 0.0050

Table 1: Accuracy (mean $\pm$ std) on the 7×7 MNIST dataset across 3 runs. Discrete-time baselines use vanilla SGD with the per-cell-best learning rate selected from a sweep over $\{10^{-4}, \ldots, 3{\cdot}10^{-1}\}$. Both models see identical training samples in identical order with batch size 1 and a single epoch.

On the more nonlinear task of distinguishing concentric circles, additional depth benefits the continuous-time model (Table 2). The two regimes again sit close to one another across all six cells, with neither systematically dominant: well-tuned SGD edges out the continuous-time network at the shallowest depth, while the continuous-time network is slightly ahead at depths 2 and 3. Across all ten comparisons in Tables 1 and 2, the differences vary by task and depth, with no consistent advantage for either regime. These results support comparable overall performance on standard, fully-supervised, in-order training.

An interesting comparison is how each regime degrades when the input–error overlap shrinks. In our continuous-time networks this happens through delays in the error signal, as analyzed in Section 4; in standard discrete-time training the analogous condition is a corresponding fraction of mislabeled data. For example, a delay of half the sample time, $\Delta = \frac{1}{2}T$ in our continuous-time model, corresponds to

Model	Train Accuracy		Test Accuracy	
	Discrete-time	Continuous-time	Discrete-time	Continuous-time
1-layer	$\mathbf{0.9670 \pm 0.0082}$	0.9241 ± 0.0133	$\mathbf{0.9658 \pm 0.0107}$	0.9470 ± 0.0101
2-layer	0.9609 ± 0.0123	$\mathbf{0.9657 \pm 0.0082}$	0.9607 ± 0.0100	$\mathbf{0.9683 \pm 0.0109}$
3-layer	0.9279 ± 0.0286	$\mathbf{0.9494 \pm 0.0588}$	0.9337 ± 0.0199	$\mathbf{0.9568 \pm 0.0452}$

Table 2: Accuracy (mean $\pm$ std) on the concentric circles dataset across 5 runs. Discrete-time baselines use vanilla SGD with the per-cell-best learning rate selected from a sweep over $\{10^{-4}, \ldots, 3{\cdot}10^{-1}\}$. Both models see identical training samples in identical order with batch size 1 and a single epoch.

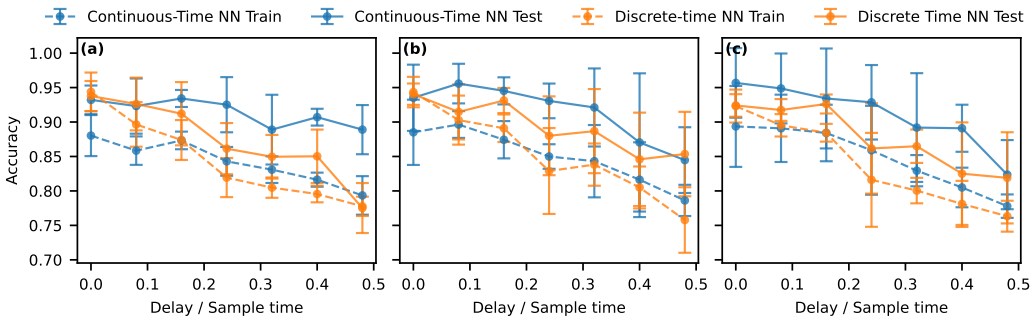


Figure 8: Delay ratio versus accuracy. Panels (a), (b), and (c) correspond to networks with 1, 2, and 3 hidden layers. Consistent with the framing of Tables 1 and 2, the two regimes have similar test accuracy at zero delay. As the delay ratio grows and input–error overlap shrinks, test accuracy generally falls in both regimes; the continuous-time model (solid blue) retains higher mean test accuracy than the discrete-time SGD baseline (solid orange) at most nonzero delays.

presenting the mismatched pairing (X_i, Y_{i-1}) and then the correct pairing (X_i, Y_i), where i is the index into the input and label datasets.

Figure 8 compares our continuous-time neural network with a standard discrete-time neural network trained with *label dithering*, which models the effect of delayed error signals without duplicating training data. Instead of splitting each sample into sub-intervals, we approximate the delay ratio $r = \tau_{\text{delay}}/T_{\text{sample}}$ by a rational fraction p/q, and then assign exactly p out of every q samples to use the *previous* label while the rest use the current label. This ensures that the fraction of "mismatched" updates matches the physical delay ratio in expectation, while keeping each sample presented only once. In this way, the training dynamics reflect the temporal overlap of input and delayed error, but the dataset size and number of SGD steps remain unchanged. The discrete-time baseline uses the same per-cell-best vanilla-SGD learning rate procedure as Tables 1 and 2, so any gap between the two curves is not attributable to a poorly tuned baseline.