Augmented Lagrangian Predictive Coding

Jeffrey Seely
Sakana AI
jeffrey@sakana.ai

Julian Gould
Sakana AI
juliangould@sakana.ai

Abstract

Predictive coding (PC) is a local-learning alternative to backpropagation (BP), training deep networks via local energy-minimization dynamics rather than a global backward pass. We introduce Augmented Lagrangian Predictive Coding (PC-ALM), which maintains PC's inference budget but aligns each weight update toward BP by accumulating per-layer constraint errors into a layer-local Lagrange multiplier. In linear PC networks, PC-ALM converges to an equilibrium with exact BP gradients distributed across the network via only layer-local updates. We analyze PC-ALM in nonlinear PC networks up to depth 128 and show that it matches BP performance across all width-depth regimes, notably in deep narrow networks where PC underperforms. PC-ALM introduces recurrent dynamics in each layer's activations. Compared to PC's heat flow on a scalar energy, PC-ALM dynamics are driven by dual ascent on the augmented Lagrangian. We observe "ballistic" credit propagation across very deep networks, with credit signals evenly distributed across layers, compared to PC's slow, diffusive credit propagation. Beyond the algorithm itself, the augmented Lagrangian framework offers a generalization of PC, and may yield insights into how distributed systems could compute and propagate BP-like credit signals through purely local dynamics.

1 Introduction

Modern deep learning relies on backpropagation (BP). Yet biological circuits cannot implement BP, at least not exactly [Lillicrap et al., 2020]. One reason is that BP is a *global* algorithm: even though BP's chain rule spatially localizes updates, the network must still create error variables in parallel with the matching forward activations, then gate weight updates in a precisely orchestrated sequence of steps. This gap has motivated research into *local* alternatives to BP, where state, error, and weight variables evolve as a single dynamical system that requires only layer-local coupling. Predictive coding (PC) is one of the more actively researched layer-local alternatives to BP [Whittington and Bogacz, 2017, Millidge et al., 2021, Salvatori et al., 2026]. In a Predictive Coding Network (PCN), a mean-squared error term is introduced between each layer, the sum of which is the PC energy. Gradient descent on this energy yields a layer-local dynamical system for updating the activation states (PC inference) and weights (PC learning). The inference equilibrium yields a different activation pattern than one would get from a forward pass, resulting in a corresponding weight update that is similar to, but not the same as, the BP gradient.

In this paper, we frame PC within a broader class of local learning methods involving the augmented Lagrangian (AL). Dual ascent on the AL—known as the method of multipliers (MM) [Hestenes, 1969]—yields a local algorithm in certain cases and has been extensively used in decentralized approaches to network training (e.g., Taylor et al. [2016]). We show that MM can be used as a replacement for PC's inference step, but with a substantial difference: at equilibrium, the Lagrange multipliers converge to exactly the gradient signal of the supervised loss.

This connection is not entirely new. LeCun [1988] observed that, at equilibrium, the multipliers of the *standard* Lagrangian encode the BP adjoints. Interestingly, the augmented Lagrangian is essentially

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

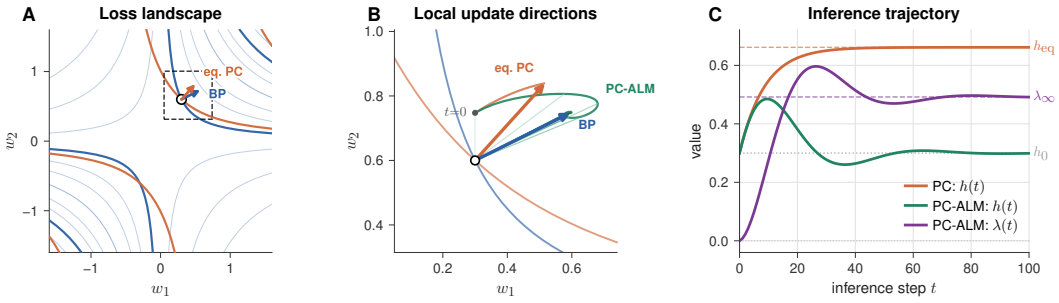


Figure 1: PC-ALM credit alignment in a scalar two-layer network ($y = w_2 w_1 x$; $h = w_1 x$; x=y=1). **(A)** BP loss landscape with BP and equilibrated-PC iso-loss curves. **(B)** Arrows are negative weight gradients for BP, equilibrated PC, and PC-ALM at different inference steps; curves trace the PC and PC-ALM gradients over inference from $t = 0$. **(C)** Inference trace. PC settles h at h_{eq}; PC-ALM returns h to $h_0 = w_1 x$ while λ converges to the negative of the BP adjoint $\lambda_\infty = (y - w_2 h_0)\, w_2$. Inference follows Algorithm 1 with $\alpha = 0.2$.

the standard Lagrangian plus the PC energy (up to the shared supervised-loss term), suggesting that there is a close relationship between PC and augmented Lagrangian methods.

Contribution: PC-ALM. The caveat of MM is that exact BP computation requires convergence to equilibrium in its inner solve, which is not guaranteed for general nonlinear networks [Wang et al., 2019]. However, we note that this difficulty is shared by PC. The PC literature handles this empirically, fixing a finite inference budget (the number of gradient steps on the PC energy) and assessing performance directly, even if inference is undersolved [Innocenti et al., 2025, Pinchetti et al., 2025]. This choice is often justified by good initializations that are close to equilibrium [Innocenti et al., 2025, Pinchetti et al., 2026]. Motivated by this, we develop a finite-inference variant of MM, called PC-ALM (for Augmented Lagrangian Method), which adds one local multiplier update to each PC inference step, so the two methods can be compared at the same inference budget.[1]

Mechanistically, PC-ALM modifies PC inference with a dual variable (i.e. Lagrange multiplier) per layer that integrates prediction errors and feeds back into the local activations. After a fixed number of inference steps, the weight update acts on a composite signal of the prediction error shifted by the dual. The recurrence introduces stable oscillatory transients on an intermediate time scale between PC's fast inference and slow weight updates.

We first show that in linear PC networks, PC-ALM converges to a solution with exact BP gradients. Our core experimental finding is that, in nonlinear networks, PC-ALM closes the performance gap between PC and BP without increasing PC's inference budget. Following Innocenti et al. [2026], we test PC-ALM at depths up to 128 on image classification. PC-ALM matches BP across all width-depth regimes, including the deep narrow networks where PC significantly underperforms BP.

We further observe experimentally that PC-ALM exhibits surprising dynamical properties. PC networks are known to concentrate credit signals near the supervised loss [Pinchetti et al., 2025, Goemaere et al., 2026]. In contrast, PC-ALM distributes credit near-uniformly across the network. Finally, we observe that the primal-dual dynamics of PC-ALM accelerate the credit-propagation wavefront in extremely deep networks, where standard PC stalls within practical inference budgets.

2 Related Work

Our current work draws on an under-explored connection between the biologically plausible deep learning literature and the distributed optimization literature.

Predictive coding. Predictive coding originated in computational neuroscience as a generative (or top-down) model of cortical inference, in which each layer predicts the activity of the layer below

[1] Code is available at `https://github.com/SakanaAI/pc-alm`.

[Rao and Ballard, 1999, Friston and Kiebel, 2009]. Whittington and Bogacz [2017] later applied the same formalism to discriminative (or bottom-up) networks, where predictions run from input to output as in a standard feedforward network. In this sense, PC provides layer-local credit assignment through inference dynamics [Millidge et al., 2021, Salvatori et al., 2026], relaxing the "locking" constraints imposed by BP's globally sequenced forward and backward passes [Jaderberg et al., 2017, Millidge et al., 2022a]. While standard PC inference uses local Jacobians, relaxed variants avoid weight transport and even omit backward activation derivatives [Millidge et al., 2020, Alonso and Neftci, 2021]. A core theme across the PC literature concerns the limitations of PC relative to BP. Pinchetti et al. [2025] recently established benchmark standards, showing that PC matches BP on small-to-medium image classification tasks but substantially underperforms in deeper networks. Beyond the vanishing/exploding gradients that also affect BP, a limitation specific to PC is the slow *credit propagation* during inference [Goemaere et al., 2026]. Innocenti et al. [2025] showed that simple tasks can be trained with PC on 100+ layer networks using a μP-style parameterization. Innocenti et al. [2026] observed that PC approximates BP in the limit of networks much wider than they are deep. Even under these parameterizations, credit in PC at a finite inference budget can remain concentrated near the output (Figure 3), indicating that the credit propagation of PC remains an important limitation. Other lines of work have focused on the benefits of PC relative to BP, notably the ability to work across arbitrary computational graphs [Salvatori et al., 2022] and the fact that letting activities settle before updating weights (termed *prospective configuration* by Song et al. 2024) yields a weight update that provides useful higher-order information compared to BP [Alonso et al., 2024, Innocenti et al., 2024].

Our work applies most directly to discriminative PC networks. PC-ALM locally computes BP-aligned credit signals, addressing both the slow credit propagation of standard PC observed by Pinchetti et al. [2025], Goemaere et al. [2026] and the PC-BP gap in deep, narrow networks observed by Innocenti et al. [2026]. At convergence, however, PC-ALM gives up the prospective configuration of PC. Nevertheless, adding a leak to the multipliers can be viewed as interpolating between the credit propagation benefits of PC-ALM and the prospective configuration benefits of PC, which we develop in Appendix E.

Augmented Lagrangian and ADMM for network training. A separate community has trained networks by the alternating direction method of multipliers (ADMM) or AL methods [Evens et al., 2021, Zhou et al., 2026], often for scalability, conditioning, or distributed optimization rather than biological plausibility. This literature is often known as "lifted" training, since the activations are considered optimization variables in addition to the network parameters (as in PC). The method of auxiliary coordinates [Carreira-Perpiñán and Wang, 2014] minimizes the same quadratic-penalty energy as PC, but increases the penalty weight to enforce the layer constraints. A related quadratic-penalty formulation also underlies ProxProp [Frerix et al., 2018]. ADMM (an extension of MM) has been used to train networks without gradient steps [Taylor et al., 2016]. For smooth activations, a variant of ADMM training provably converges to a KKT point at rate $O(1/k)$ [Zeng et al., 2021]. Wang et al. [2025b] use the AL with multipliers on every operation to train RNNs, exploiting ReLU's piecewise-linear structure to obtain closed-form block updates; applied to a feedforward MLP, their method is closely related to PC-ALM. A set of lifted training methods [Askari et al., 2018, Li et al., 2018, Gu et al., 2020, Zach and Estellers, 2019, Wang and Benning, 2023, Wang et al., 2025a] replace hard layer constraints with activation-specific penalties or relaxations (proximal, Fenchel/Lagrange-relaxation, contrastive, Bregman), but do not use layer multipliers. Our work differs algorithmically from this literature in that we study undersolved inference and focus less on proving convergence guarantees; this allows direct comparison to PC under its experimental standards.

3 Background

We start by reviewing the constrained optimization formalism for training a deep feedforward network.

Constrained network. Consider a feedforward network of depth L with weights $\theta = \{W_i\}_{i=1}^{L}$ and elementwise nonlinearity σ (e.g., identity, tanh, ReLU). Given input $x \in \mathbb{R}^{n_0}$ and target $y \in \mathbb{R}^{n_L}$, the forward pass specifies $h_i = \sigma(W_i h_{i-1})$ from $h_0 := x$ for $i = 1, \dots, L-1$, with linear readout

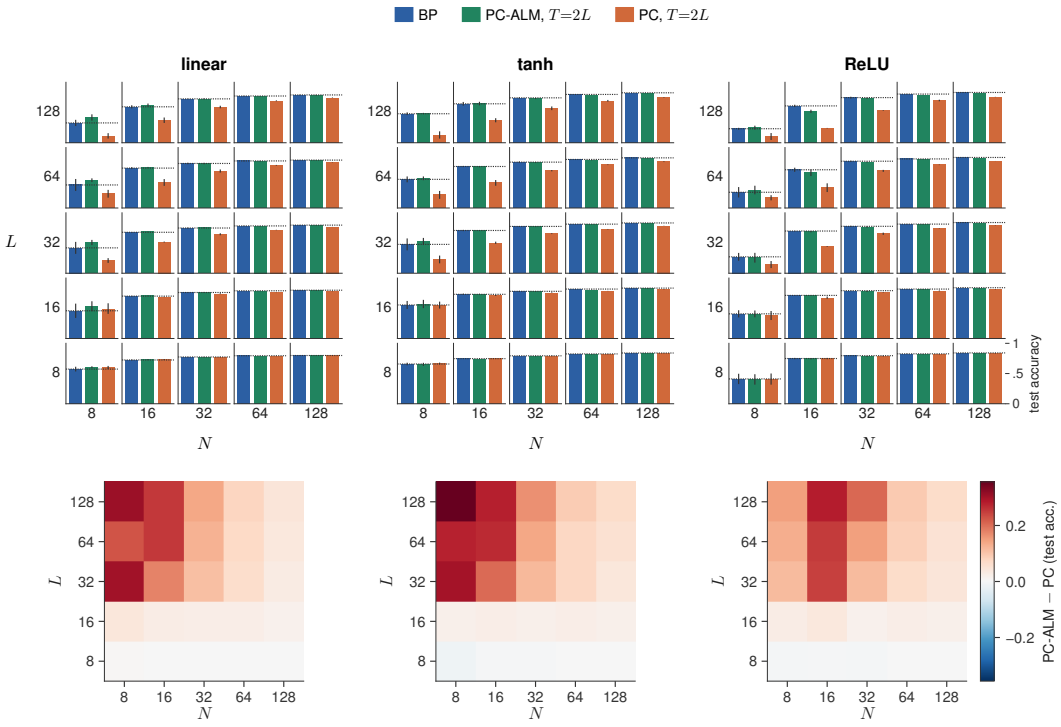


Figure 2: Width-depth results on Fashion-MNIST. PC-ALM matches BP at a budget of $T = 2L$ across the full range of width N, depth L, and activations. PC-ALM outperforms PC in deep narrow networks.

$W_L h_{L-1}$. We can write the optimization problem as

$$\begin{aligned} \min_{\theta,\, h} \quad & \tfrac{1}{2}\left\|y - W_L h_{L-1}\right\|^2 \\ \text{s.t.} \quad & h_i = \sigma(W_i h_{i-1}), \quad i = 1, \ldots, L-1, \end{aligned} \tag{1}$$

where the activations $h = \{h_i\}_{i=1}^{L-1}$ are optimization variables alongside the weights θ.

Predictive coding as quadratic penalty. The penalty relaxation of (1) replaces each layer constraint with a squared-error term, giving the PC energy [Whittington and Bogacz, 2017, Millidge et al., 2022b, Salvatori et al., 2022]:

$$\mathcal{F}_{\text{PC}}(h, \theta) \;=\; \tfrac{1}{2}\left\|y - W_L h_{L-1}\right\|^2 \;+\; \frac{\rho}{2} \sum_{i=1}^{L-1} \left\|h_i - \sigma(W_i h_{i-1})\right\|^2, \tag{2}$$

for a penalty strength ρ.

Equilibrated PC. For each sample (x, y), PC alternates between two steps:

$$\text{inference:} \quad h^\star \;=\; \arg\min_h \mathcal{F}_{\text{PC}}(h, \theta), \tag{3a}$$

$$\text{learning:} \quad \theta \;\leftarrow\; \theta - \eta_\theta \nabla_\theta \mathcal{F}_{\text{PC}}(h^\star, \theta). \tag{3b}$$

The inference step replaces a standard deep network's forward and backward passes. The alternating steps are meant to reflect fast (inference) and slow (learning) dynamics of the underlying system.

Finite-inference PC. As written, the argmin in (3a) globally couples all layer activations. However, gradient descent on $\mathcal{F}_{\text{PC}}$ decouples into layer-local updates:

$$h_i(t+1) \;=\; h_i(t) \;-\; \eta_h \nabla_{h_i} \mathcal{F}_{\text{PC}}(h(t), \theta), \quad i = 1, \ldots, L-1, \quad t = 0, \ldots, T-1. \tag{4}$$

Each h_i appears in only two terms of $\mathcal{F}_{\text{PC}}$, so $\nabla_{h_i}\mathcal{F}_{\text{PC}}$ depends only on $h_{i-1}(t), h_i(t), h_{i+1}(t)$ and the adjacent weights W_i, W_{i+1}. Activations are typically initialized from their forward-pass values. In general, the number of inference steps required to reach convergence depends on the architecture [Alonso et al., 2024]; in practice, the iteration is typically run either to numerical convergence [Innocenti et al., 2024] or for a fixed T-step budget [Innocenti et al., 2025, Pinchetti et al., 2025]. We refer to this as finite-inference PC to distinguish it from the idealized equilibrated case.

Augmented Lagrangian. Attaching multipliers $\lambda = \{\lambda_i\}_{i=1}^{L-1}$ to the constraints of (1) and keeping the quadratic penalty gives the augmented Lagrangian [Hestenes, 1969, Powell, 1969, Bertsekas, 1976]:

$$\mathcal{L}_\rho(h,\theta,\lambda) \;=\; \tfrac{1}{2}\big\|y - W_L h_{L-1}\big\|^2 \;+\; \sum_{i=1}^{L-1} \lambda_i^\top \big(h_i - \sigma(W_i h_{i-1})\big) \;+\; \frac{\rho}{2}\sum_{i=1}^{L-1}\big\|h_i - \sigma(W_i h_{i-1})\big\|^2. \quad (5)$$

Setting $\lambda \equiv 0$ recovers the PC energy: $\mathcal{F}_{\text{PC}}(h,\theta) = \mathcal{L}_\rho(h,\theta,0)$. Setting $\rho = 0$ recovers the standard Lagrangian of (1).

Method of multipliers. Returning to the constrained optimization problem in (1), we can solve for (h,λ) by performing dual ascent on the AL (the method of multipliers [Hestenes, 1969, Powell, 1969, Bertsekas, 1976]). For each sample, initialize $\lambda_i \equiv 0$ and alternate between primal and dual updates

$$\text{primal:} \quad h \;\leftarrow\; \arg\min_h \mathcal{L}_\rho(h,\theta,\lambda), \quad (6a)$$

$$\text{dual:} \quad \lambda_i \;\leftarrow\; \lambda_i + \rho\big(h_i - \sigma(W_i h_{i-1})\big), \quad i = 1,\dots,L-1, \quad (6b)$$

for $k = 1,\dots,K$ outer rounds or until convergence criteria are met. Under standard regularity conditions, these updates converge for sufficiently large ρ [Bertsekas, 1976]. The method of multipliers here is being used to solve the "inference" step but on the *constrained* problem (1), not PC's quadratic relaxation (2). Therefore, at convergence to a KKT point $(h^\star, \lambda^\star)$, the activations $h^\star$ are simply what one would get from a forward pass. However, the converged multipliers $\lambda^\star$ represent meaningful local error signals, yielding a weight update:

$$\text{learning:} \quad \theta \;\leftarrow\; \theta - \eta_\theta \nabla_\theta \mathcal{L}_\rho(h^\star,\theta,\lambda^\star). \quad (7)$$

This corresponds exactly to a BP update, as expanded below.

Multipliers as backpropagation gradients. LeCun [1988] observed that at any KKT point of (1), the multipliers of the standard Lagrangian ((5) with $\rho = 0$) encode the backpropagation adjoints. Namely, with h fixed at the forward-pass values, back-substitution from the output recovers λ in closed form, with λ equal to the negative BP adjoint at every layer (Appendix A).

Local algorithm via the augmented Lagrangian. This identification suggests an iterative route to the BP adjoints by dual ascent on the standard Lagrangian, which decomposes layerwise (a special case of dual decomposition). On a deep network's nonconvex Lagrangian, however, the iteration can diverge with λ unbounded [Nocedal and Wright, 2006]. The augmented Lagrangian (5) stabilizes the dual update without altering the KKT solution, but at the cost of breaking the layerwise decomposition; the primal argmin (6a) now couples all activations globally. The optimization literature typically resolves this with ADMM [Boyd et al., 2011]. We point out that solving the primal argmin with gradient descent, as in PC, also preserves layer-locality, since $\nabla_h \mathcal{L}_\rho$ at layer i depends only on adjacent layers' activations and multipliers. Therefore, pairing PC-like primal updates with dual updates suggests a route to layer-local learning with gradients more aligned with BP than PC's.

4 Method: PC-ALM

The method of multipliers (6) wraps an outer dual loop around the inner argmin, which must be solved at every outer round. Reaching this inner stationarity is the central bottleneck of PC inference in practice [Pinchetti et al., 2025], and is inherited by MM. We therefore develop PC-ALM, a finite-inference variant of MM that uses the same inference budget T as finite-inference PC. PC-ALM replaces the argmin in (6a) with one gradient descent step on $\mathcal{L}_\rho$, applies the dual update immediately after, and loops $T-1$ times before a final primal step. The full procedure is Algorithm 1.

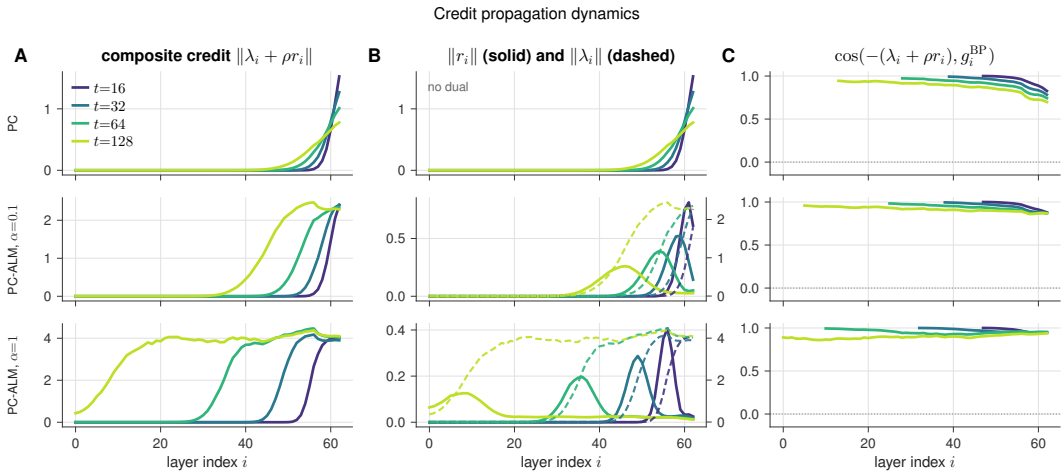


Figure 3: Diffusive versus ballistic credit propagation. ReLU residual MLP with $N = 32$, $L = 64$ at initialization, on a random Gaussian input-target pair. Panel B shows that the prediction error r_i (left axis scale) leads the dual λ_i (right axis scale) across the network. PC-ALM enables faster and more even credit propagation across the network compared to PC.

Algorithm 1 PC-ALM training

given dataset D, penalty ρ, inference steps T, step sizes $\eta_h, \alpha, \eta_\theta$
for each batch $B \subset D$ **do**
  for each sample $(x, y) \in B$ in parallel **do**
    $h_0 \leftarrow x$; $h_i \leftarrow \sigma(W_i h_{i-1})$ for $i = 1, \dots, L-1$ ▷ forward init
    $\lambda_i \leftarrow 0$ for $i = 1, \dots, L-1$
    for $t = 1, \dots, T-1$ **do**
      $h_i \leftarrow h_i - \eta_h \nabla_{h_i} \mathcal{L}_\rho(h, \theta, \lambda)$ for $i = 1, \dots, L-1$ ▷ primal step
      $\lambda_i \leftarrow \lambda_i + \alpha\,(h_i - \sigma(W_i h_{i-1}))$ for $i = 1, \dots, L-1$ ▷ dual step
    end for
    $h_i \leftarrow h_i - \eta_h \nabla_{h_i} \mathcal{L}_\rho(h, \theta, \lambda)$ for $i = 1, \dots, L-1$ ▷ final primal step
  end for
  $W_i \leftarrow W_i - \eta_\theta \frac{1}{|B|} \sum_{(x,y)\in B} \nabla_{W_i} \mathcal{L}_\rho(h, \theta, \lambda)$ for $i = 1, \dots, L$ ▷ learning step
end for

Inference gradients. To analyze the difference between $\mathcal{F}_{\mathrm{PC}}$ and $\mathcal{L}_\rho$, we rewrite $\mathcal{L}_\rho$ in a more interpretable form. Let $r_i := h_i - \sigma(W_i h_{i-1})$ denote the residual (or prediction error) at layer i. Completing the square on each cross term in (5),

$$\lambda_i^\top r_i + \tfrac{\rho}{2}\|r_i\|^2 = \tfrac{\rho}{2}\left\|h_i - \big(\sigma(W_i h_{i-1}) - \lambda_i/\rho\big)\right\|^2 - \tfrac{1}{2\rho}\|\lambda_i\|^2. \tag{8}$$

Substituting (8) back into (5),

$$\mathcal{L}_\rho(h, \theta, \lambda) = \tfrac{1}{2}\|y - W_L h_{L-1}\|^2 + \frac{\rho}{2}\sum_{i=1}^{L-1}\left\|h_i - \big(\sigma(W_i h_{i-1}) - \lambda_i/\rho\big)\right\|^2 - \frac{1}{2\rho}\sum_{i=1}^{L-1}\|\lambda_i\|^2. \tag{9}$$

The last term has no h-dependence and drops out of $\nabla_h \mathcal{L}_\rho$. The first two terms are the PC free energy (2) with each prediction target shifted from $\sigma(W_i h_{i-1})$ to $\sigma(W_i h_{i-1}) - \lambda_i/\rho$. Differentiating with respect to h_i for $1 \le i \le L-2$,

$$\nabla_{h_i} \mathcal{F}_{\mathrm{PC}} = \rho\, r_i - W_{i+1}^\top \operatorname{diag}\big(\sigma'(W_{i+1} h_i)\big)\, \rho\, r_{i+1}, \tag{10}$$

$$\nabla_{h_i} \mathcal{L}_\rho = (\rho\, r_i + \lambda_i) - W_{i+1}^\top \operatorname{diag}\big(\sigma'(W_{i+1} h_i)\big)\, (\rho\, r_{i+1} + \lambda_{i+1}). \tag{11}$$

In PC-ALM, each prediction target is shifted by $-\lambda_i/\rho$ relative to PC.

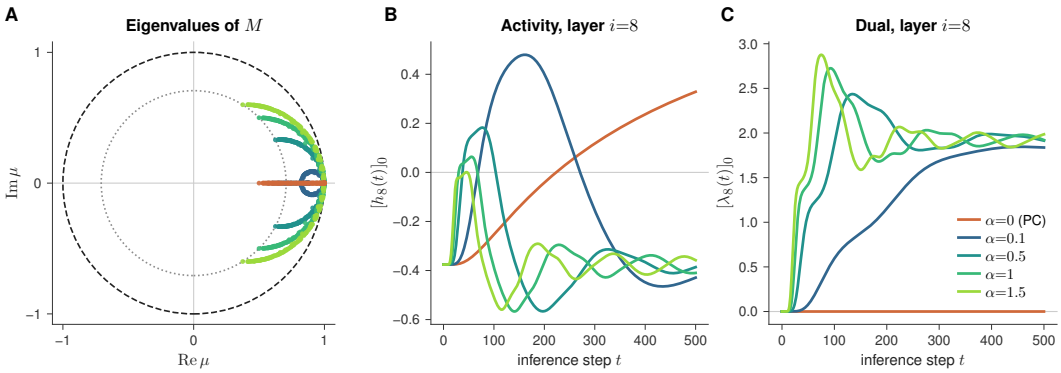


Figure 4: Linear PC-ALM inference dynamics with $L = 16$, $N = 8$. **(A)** Eigenvalues of the block iteration matrix M (Equation (31)). The inner dotted circle indicates a radius of $\sqrt{1 - \eta_h \rho\, \sigma_{\max}(A)^2}$. **(B, C)** Primal and dual neuron activity traces.

Dual step. After each activity step, the multiplier accumulates the residual:

$$\lambda_i \;\leftarrow\; \lambda_i \;+\; \alpha\, r_i, \qquad i = 1, \ldots, L-1. \tag{12}$$

The classical MM update (6b) uses rate ρ with the activity solved to stationarity. PC-ALM takes a single inner gradient step instead, so r_i carries unsettled primal gradient on top of constraint infeasibility; we use a separate hyperparameter α for the dual rate. $\alpha = 0$ recovers PC; $\alpha = \rho$ with the inner argmin solved recovers MM.

Weight update. After $T - 1$ primal-dual cycles and a final primal step, weights are updated:

$$\theta \;\leftarrow\; \theta - \eta_\theta\, \nabla_\theta \mathcal{L}_\rho(h, \theta, \lambda). \tag{13}$$

The composite signal $\lambda_i + \rho r_i$ at the weight gradient has λ_i from the integrated dual history and ρr_i from the residual at the final h. By (9), this is the PC weight update with each prediction target shifted by $-\lambda_i/\rho$.

4.1 Interpretation

PC-ALM augments each layer with a per-layer Lagrange multiplier λ_i of the same shape as h_i. On a single sample, the activity loop $t = 0, \ldots, T-1$ initializes h at the forward-pass values and $\lambda \equiv 0$. The first activity step is therefore a standard PC step; with $\lambda \equiv 0$, (11) reduces to (10). The first dual step then injects the layer-i residual r_i into λ_i. From the second activity step onward, the activity gradient at layer i sees the augmented signal $\rho\, r_i + \lambda_i$ in place of $\rho\, r_i$ alone—a lagged correction entering via the integrated multiplier from prior steps.

If inference converges, the activation h_i returns to its forward-pass value (the constraint $h_i = \sigma(W_i h_{i-1})$ is restored, $r_i \to 0$). Across the inference loop, λ_i accumulates whatever pressure the layer-i constraint exerts on adjacent activations, pushing h_i back toward its forward-pass value, and at the fixed point this accumulated pressure is exactly the negative BP adjoint at that layer (Appendix A).

Figure 1 illustrates this on a scalar two-layer example. At a fixed weight where equilibrated PC and BP disagree, the dual integrates to the negative BP adjoint $\lambda_\infty = (y - w_2 h_0)\, w_2$ while the activation returns to its forward-pass value $h_0 = w_1 x$. The PC-ALM weight update therefore rotates onto BP.

5 Results

We now present a series of theoretical and empirical results for PC-ALM.

5.1 Linear PC-ALM computes exact BP gradients at equilibrium

PC-ALM is an inexact formulation of MM since each primal step is a single gradient step instead of the full argmin. Nevertheless, we show that in a linear PC network, under stability conditions,

PC-ALM converges precisely to the KKT point of the constrained optimization problem (Appendix C, Proposition 3). Thus, in linear PCNs, PC-ALM is a layer-local dynamical system that distributes exact BP gradient signals across the network.

5.2 Stability and oscillatory modes

We also provide stability conditions for PC-ALM. The linear PC-ALM dynamics are governed by an iteration matrix M (Equation (31)) and stability requires that its spectral radius be less than 1 (i.e., all eigenvalues inside the open unit disc). Dropping the supervised-loss block ($B = 0$), this reduces to the constraint-only modewise condition

$$\underbrace{\eta_h \, \sigma_i^2 \, 2\rho < 4}_{\text{PC}} \qquad \underbrace{\eta_h \, \sigma_i^2 \, (2\rho + \alpha) < 4}_{\text{PC-ALM}} \tag{14}$$

at every singular mode σ_i of the constraint operator A (Appendix C, Corollary 1). For $\alpha = 0$ we recover the stability condition for PC. The mean-field parameterizations we use (Appendix B) stabilize $\sigma_{\max}$ over a wide range of network depths and widths. Setting $\rho = 1$, we roughly obtain $\eta_{\max} \approx 0.5$ and $\alpha_{\max} \approx 2$.

The oscillatory modes of PC-ALM are shown in Figure 4. Notably, in standard PC, all activation dynamics are gradient flows and do not produce oscillatory transients (e.g., eigenvalues on the real line in Figure 4 for $\alpha = 0$). The complex eigenvalues of PC-ALM lie within an annulus with inner radius $\sqrt{1 - \eta_h \, \rho \, \sigma_{\max}(A)^2}$ and outer radius 1. The dual rate α slides them along this circle, setting oscillation frequency.

Importantly, convergence to the KKT solution and the BP gradients it provides does not depend on the specific stable choice of η_h and α; these values only shape the transient dynamics on the path to equilibrium.

5.3 PC-ALM closes the PC-BP gap in deep narrow networks

Beyond the linear analysis, we test PC-ALM experimentally in nonlinear PCNs. We sweep $(N, L) \in \{8, 16, 32, 64, 128\}^2$ in residual MLPs under the mean-field parameterization of Innocenti et al. [2026] (details in Appendix G), comparing PC, PC-ALM, and BP at matched inference budget on Fashion-MNIST and MNIST with identity, tanh, and ReLU activations. PC performance degrades along the depth axis, reproducing the deep-narrow failure mode of Innocenti et al. [2026]. PC-ALM recovers a fraction of the BP gap at a budget of $T = L$ (Figure 11) while closing the gap entirely at a budget of $T = 2L$ across the full range of N, L, and activations (Figure 2). Further experimental details are provided in Appendix G. On ResNet-18 with CIFAR-10 and Tiny ImageNet, PC-ALM improves on PC but remains below BP (Appendix G.1).

5.4 PC-ALM is robust to reparameterization

We investigate whether the performance of PC-ALM is due to the choice of mean-field parameterization, which stabilizes both PC and BP. Holding $(N, L) = (32, 64)$, we sweep two parameterization probes around the mean-field endpoint of Innocenti et al. [2026]: a standard parameterization to mean-field interpolation $\lambda_{\text{sp}} \in \{0, 0.25, 0.5, 0.75, 1\}$ at $\gamma_0 = 1$, and a lazy-to-rich axis $\gamma_0 \in \{0.1, 0.5, 1, 2, 3, 4\}$ at $\lambda_{\text{sp}} = 1$. Across all values, PC-ALM matches the performance of BP (Figs. 6, 7), showing that it remains stable and robust even across unfavorable parameterization regimes.

5.5 Ballistic credit propagation wavefront

In addition to the experimental verification above, we note further surprising dynamical properties of PC-ALM. PCNs are known to exhibit signal decay across layers [Pinchetti et al., 2025], which is exponential in depth [Goemaere et al., 2026]. We reproduce this result as shown in Figure 3 (top). Surprisingly, the primal-dual dynamics of PC-ALM accelerate the credit propagation wavefront across the network until the composite signal $\lambda_i + \rho r_i$ distributes evenly. The prediction-error component r_i propagates ahead of the lagged dual λ_i (Figure 3B). Appendix D gives the linear wavefront calculation. We call this ballistic credit propagation in contrast to PC's diffusive propagation.

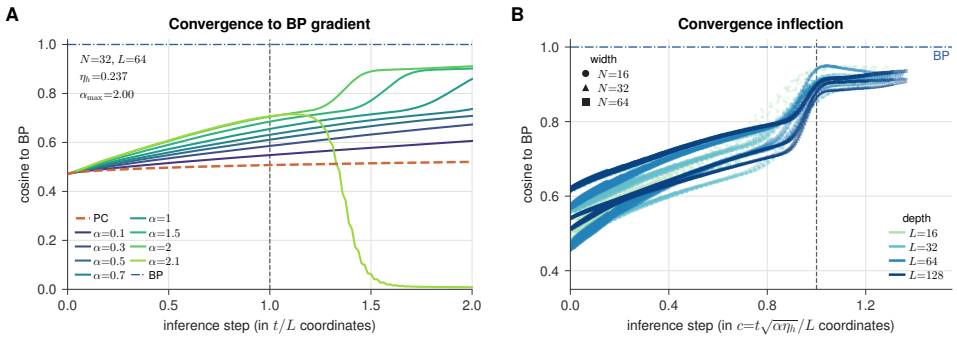


Figure 5: Cosine between the PC-ALM weight gradient and the BP weight gradient at an initialized network. **(A)** Alignment as a function of iteration, normalized to L. **(B)** Alignment as a function of iteration t, in units of $c = t\sqrt{\alpha\,\eta_h}/L$ across different depths and widths.

5.6 PC-ALM's BP alignment exhibits an inflection at a predictable inference step

We observe that PC-ALM provides a modest-but-measurable boost to PC at low inference budgets, but that this boost "snaps" to stronger BP alignment at inference steps of around twice the network depth when using default hyperparameter $\alpha = 1$. This phenomenon is illustrated in Figure 5A, where the BP alignment yields an inflection point whose location is controlled by α. Higher α pulls the inflection to an earlier inference step. We found this phenomenon to be present for all depths and widths. Further, the location of the inflection can be predicted across a wide range of network depths and widths, namely, at $t \approx L/\sqrt{\alpha\,\eta_h}$ (Figure 5B), with arguments developed in Appendix D.

5.7 PC-ALM is inference efficient

From a pragmatic standpoint, PC-ALM doubles PC's activation memory cost, albeit with modest additional compute. Under mean-field scaling, Innocenti et al. [2026] show that increasing width aligns PC's gradient closer to BP, giving two routes to BP alignment: widening PC, or running PC-ALM at fixed width. To achieve a 0.9 cosine similarity to BP, PC-ALM is more efficient by an order of magnitude. Results are in Figure 9, with the argument developed in Appendix F.

5.8 Dependence on initialization

Both PC and PC-ALM degrade when the initial activities are displaced from the forward pass (Appendix H). PC-ALM stays ahead of PC up to a displacement of 0.3 times the RMS activity, and falls behind only at larger displacements, where both are far below BP. Far from the KKT point, the multipliers integrate large residuals unrelated to the output loss which can degrade performance worse than PC.

5.9 Leaky multipliers interpolate between PC and PC-ALM

Adding a leak κ to the multiplier update, $\lambda_i \leftarrow (1-\kappa)\lambda_i + \alpha r_i$, interpolates between PC-ALM ($\kappa \to 0$) and PC ($\alpha/\kappa \to 0$) (Appendix E). It therefore also interpolates between the benefits of each, namely prospective configuration for PC and credit propagation and BP alignment for PC-ALM. The resulting Pareto front bows toward the upper right, so intermediate leaks recover much of PC's prospective configuration while keeping most of PC-ALM's BP alignment (Figure 8).

6 Discussion

We introduce PC-ALM, a generalization of finite-inference PC that arises as finite-inference dual ascent on the augmented Lagrangian of a feedforward neural network. PC and PC-ALM address two different optimization problems—the quadratic relaxation in PC's case, and the original constrained problem in PC-ALM's. They share the same inference budget aside from PC-ALM's dual update, but exhibit distinct dynamical properties. We verified that PC-ALM is stable for linear PCNs and

provides exact BP-aligned credit signals across the network through layer-local accumulation of prediction errors. In nonlinear networks, we further found that PC-ALM is robust and matches BP performance, notably in parameter regimes where PC and BP diverge. We expand on current limitations of our work in Appendix I.

6.1 Connection to other Lagrangian-based methods

PC-ALM is one of several training methods that begin by writing feedforward training as a constrained problem $\min \texttt{Loss}(h_L, y)$ subject to $h_\ell = \sigma(W_\ell h_{\ell-1})$, and attaching Lagrange multipliers and/or quadratic penalties to the layer constraints. Methods within this family differ in (i) which terms of the augmented Lagrangian they retain, (ii) how they solve the inner activity subproblem, and (iii) how the multipliers, if any, are recovered. Standard PC relies only on the penalty.

Like PC, the BlockProp algorithm [Gotmare et al., 2018] works with a quadratic penalty for a different motivation—model-parallel training across blocks of layers. The authors use an L2 gluing penalty and note that the Taylor-style Lagrange multiplier was unnecessary and less stable in their experiments. They further show that BlockProp with single-layer blocks and a quadratic gluing energy is equivalent to Equilibrium Propagation (EP) of Scellier and Bengio [2017]. EP itself takes a different route around the dual-ascent instability. Rather than iterating the multipliers, it recovers them implicitly by *nudging*—a finite-difference method on the energy gradient between a free fixed point ($\beta = 0$, in EP notation) and a weakly-clamped fixed point ($\beta \to 0^+$). This approach yields contrastive Hebbian updates that estimate $\lambda^\star$ without ever carrying it as state. EP therefore needs two fixed-point phases; PC-ALM runs primal and dual updates in a single loop. Dual Propagation [Høier et al., 2023] extends the EP nudging idea to a single-phase dyadic-neuron scheme, where the per-neuron difference between nudged states plays a role structurally analogous to PC-ALM's multiplier. The recent work of Scurria [2026] resembles PC-ALM but via the standard Lagrangian. We expand on further related work in Appendix J.

Acknowledgments and Disclosure of Funding

We thank the Sakana AI research team for their feedback. This work has been done at Sakana AI without any other source of funding. The authors declare no competing interests.

References

Nicholas Alonso and Emre Neftci. Tightening the biological constraints on gradient-based predictive coding. In *International Conference on Neuromorphic Systems 2021*, pages 1–9, 2021. doi: 10.1145/3477145.3477148.

Nicholas Alonso, Jeffrey L. Krichmar, and Emre Neftci. Understanding and improving optimization in predictive coding networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 10812–10820, 2024. doi: 10.1609/aaai.v38i10.28954.

Armin Askari, Geoffrey Negiar, Rajiv Sambharya, and Laurent El Ghaoui. Lifted neural networks. *arXiv preprint arXiv:1805.01532*, 2018.

Yoshua Bengio. How auto-encoders could provide credit assignment in deep networks via target propagation. *arXiv preprint arXiv:1407.7906*, 2014.

Dimitri P. Bertsekas. Multiplier methods: A survey. *Automatica*, 12(2):133–145, 1976.

Vicente Bosca and Robert Ghrist. Neural networks as local-to-global computations. *arXiv preprint arXiv:2603.14831*, 2026.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011. doi: 10.1561/2200000016.

Colin Bredenberg, Ezekiel Williams, Cristina Savin, Blake Richards, and Guillaume Lajoie. Formalizing locality for normative synaptic plasticity models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 5653–5684. Curran Associates,

Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/120339238f293d4ae53a7167403abc4b-Paper-Conference.pdf.

Miguel Á. Carreira-Perpiñán and Weiran Wang. Distributed optimization of deeply nested systems. In *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics, PMLR 33*, 2014.

Brecht Evens, Puya Latafat, Andreas Themelis, Johan Suykens, and Panagiotis Patrinos. Neural network training as an optimal control problem: An augmented Lagrangian approach. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 5136–5143. IEEE, 2021. doi: 10.1109/CDC45484.2021.9682842.

Thomas Frerix, Thomas Möllenhoff, Michael Moeller, and Daniel Cremers. Proximal backpropagation. In *International Conference on Learning Representations*, 2018. URL https://arxiv.org/abs/1706.04638.

Karl Friston and Stefan Kiebel. Predictive coding under the free-energy principle. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 364(1521):1211–1221, 2009. doi: 10.1098/rstb.2008.0300.

Cédric Goemaere, Gaspard Oliviers, Rafal Bogacz, and Thomas Demeester. ePC: Fast and deep predictive coding in digital simulation. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 35407–35442. PMLR, 2026.

Akhilesh Gotmare, Valentin Thomas, Johanni Brea, and Martin Jaggi. Decoupling backpropagation using constrained optimization methods. In *ICML 2018 Workshop on Credit Assignment in Deep Learning and Deep Reinforcement Learning*, 2018. URL https://openreview.net/forum?id=BygR79WfWm.

Fangda Gu, Armin Askari, and Laurent El Ghaoui. Fenchel lifted networks: A Lagrange relaxation of neural network training. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics, PMLR 108*, 2020.

Jakob Hansen and Robert Ghrist. Distributed optimization with sheaf homological constraints. In *2019 57th Annual Allerton Conference on Communication, Control, and Computing*, pages 766–773, 2019. doi: 10.1109/ALLERTON.2019.8919796.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

Magnus R. Hestenes. Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 4:303–320, 1969.

Rasmus Høier, D. Staudt, and Christopher Zach. Dual propagation: Accelerating contrastive Hebbian learning with dyadic neurons. In *International Conference on Machine Learning, PMLR 202*, 2023.

Francesco Innocenti, El Mehdi Achour, Ryan Singh, and Christopher L. Buckley. Only strict saddles in the energy landscape of predictive coding networks? *arXiv preprint arXiv:2408.11979*, 2024.

Francesco Innocenti, El Mehdi Achour, and Christopher L. Buckley. μPC: Scaling predictive coding to 100+ layer networks. *arXiv preprint arXiv:2505.13124*, 2025.

Francesco Innocenti, El Mehdi Achour, and Rafal Bogacz. On the infinite width and depth limits of predictive coding networks. *arXiv preprint arXiv:2602.07697*, 2026.

Max Jaderberg, Wojciech Marian Czarnecki, Simon Osindero, Oriol Vinyals, Alex Graves, David Silver, and Koray Kavukcuoglu. Decoupled neural interfaces using synthetic gradients. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1627–1635, 2017.

Jayash Koshal, Angelia Nedić, and Uday V. Shanbhag. Multiuser optimization: Distributed algorithms and error analysis. *SIAM Journal on Optimization*, 21(3):1046–1081, 2011. doi: 10.1137/090770102.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Ya Le and Xuan Yang. Tiny ImageNet visual recognition challenge. Technical report, Stanford University, CS 231N, 2015.

Yann LeCun. A theoretical framework for back-propagation. In *Proceedings of the 1988 Connectionist Models Summer School*, pages 21–28, 1988.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Dong-Hyun Lee, Saizheng Zhang, Asja Fischer, and Yoshua Bengio. Difference target propagation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 498–515. Springer, 2015.

Jia Li, Cong Fang, and Zhouchen Lin. Lifted proximal operator machines. *arXiv preprint arXiv:1811.01501*, 2018.

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. doi: 10.1038/s41583-020-0277-3.

Beren Millidge, Alexander Tschantz, Anil Seth, and Christopher L. Buckley. Relaxing the constraints on predictive coding models. *arXiv preprint arXiv:2010.01047*, 2020.

Beren Millidge, Anil Seth, and Christopher L. Buckley. Predictive coding: A theoretical and experimental review. *arXiv preprint arXiv:2107.12979*, 2021.

Beren Millidge, Tommaso Salvatori, Yuhang Song, Rafal Bogacz, and Thomas Lukasiewicz. Predictive coding: Towards a future of deep learning beyond backpropagation? In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, pages 5538–5545, 2022a.

Beren Millidge, Yuhang Song, Tommaso Salvatori, Thomas Lukasiewicz, and Rafal Bogacz. A theoretical framework for inference and learning in predictive coding networks. *arXiv preprint arXiv:2207.12316*, 2022b.

Beren Millidge, Alexander Tschantz, and Christopher L. Buckley. Predictive coding approximates backprop along arbitrary computation graphs. *Neural Computation*, 34(6):1329–1368, 2022c. doi: 10.1162/neco_a_01497.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, 2nd edition, 2006.

Gaspard Oliviers, Elene Lominadze, and Rafal Bogacz. Understanding sample efficiency in predictive coding. *arXiv preprint arXiv:2605.11911*, 2026.

Luca Pinchetti, Chang Qi, Oleh Lokshyn, Gaspard Oliviers, Cornelius Emde, Mufeng Tang, Amine M'Charrak, Simon Frieder, Bayar Menzat, Rafal Bogacz, Thomas Lukasiewicz, and Tommaso Salvatori. Benchmarking predictive coding networks – made simple. *arXiv preprint arXiv:2407.01163*, 2025.

Luca Pinchetti, Simon Frieder, Thomas Lukasiewicz, and Tommaso Salvatori. Faster predictive coding networks via better initialization. *arXiv preprint arXiv:2601.20895*, 2026.

Michael J. D. Powell. A method for nonlinear constraints in minimization problems. In R. Fletcher, editor, *Optimization*, pages 283–298. Academic Press, 1969.

Rajesh P. Rao and Dana H. Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2:79–87, 1999. doi: 10.1038/4580.

Robert Rosenbaum. On the relationship between predictive coding and backpropagation. *PLoS ONE*, 17(3):e0266102, 2022. doi: 10.1371/journal.pone.0266102.

Tommaso Salvatori, Luca Pinchetti, Beren Millidge, Yuhang Song, Tianyi Bao, Rafal Bogacz, and Thomas Lukasiewicz. Learning on arbitrary graph topologies via predictive coding. In *Advances in Neural Information Processing Systems*, 2022.

Tommaso Salvatori, Ankur Mali, Christopher L. Buckley, Thomas Lukasiewicz, Rajesh P. N. Rao, Karl Friston, and Alexander Ororbia. A survey on neuro-mimetic deep learning via predictive coding. *Neural Networks*, 195:108161, 2026. doi: 10.1016/j.neunet.2025.108161.

Benjamin Scellier and Yoshua Bengio. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in Computational Neuroscience*, 11:24, 2017. doi: 10.3389/fncom.2017.00024.

Antonino Emanuele Scurria. A physical theory of backpropagation: Exact gradients from the least-action principle. *arXiv preprint arXiv:2602.02281*, 2026.

Jeffrey Seely. Sheaf cohomology of linear predictive coding networks. *arXiv preprint arXiv:2511.11092*, 2025.

Yuhang Song, Beren Millidge, Tommaso Salvatori, Thomas Lukasiewicz, Zhenghua Xu, and Rafal Bogacz. Inferring neural activity before plasticity as a foundation for learning beyond backpropagation. *Nature Neuroscience*, 2024. doi: 10.1038/s41593-023-01514-1.

Yujie Tang, Emiliano Dall'Anese, Andrey Bernstein, and Steven Low. Running primal-dual gradient method for time-varying nonconvex problems. *arXiv preprint arXiv:1812.00613*, 2018.

Gavin Taylor, Ryan Burmeister, Zheng Xu, Bharat Singh, Ankit Patel, and Tom Goldstein. Training neural networks without gradients: A scalable ADMM approach. In *Proceedings of the 33rd International Conference on Machine Learning, PMLR 48*, 2016.

Junxiang Wang, Fuxun Yu, Xiang Chen, and Liang Zhao. ADMM for efficient deep learning with global convergence. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 111–119, 2019. doi: 10.1145/3292500.3330936.

Xiaoyu Wang and Martin Benning. Lifted Bregman training of neural networks. *Journal of Machine Learning Research*, 24, 2023.

Xiaoyu Wang, Alexandra Valavanis, Azhir Mahmood, Andreas Mang, Martin Benning, and Audrey Repetti. A unified framework for lifted training and inversion approaches. *arXiv preprint arXiv:2510.09796*, 2025a.

Yue Wang, Chao Zhang, and Xiaojun Chen. An augmented Lagrangian method for training recurrent neural networks. *SIAM Journal on Scientific Computing*, 47(1):C22–C51, 2025b. doi: 10.1137/23M1627614.

James C. R. Whittington and Rafal Bogacz. An approximation of the error backpropagation algorithm in a predictive coding network with local Hebbian synaptic plasticity. *Neural Computation*, 29(5): 1229–1262, 2017. doi: 10.1162/neco_a_00949.

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–19, 2018.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Christopher Zach and Virginia Estellers. Contrastive learning for lifted networks. *arXiv preprint arXiv:1905.02507*, 2019.

Jinshan Zeng, Shao-Bo Lin, Yuan Yao, and Ding-Xuan Zhou. On ADMM in deep learning: Convergence and saturation-avoidance. *Journal of Machine Learning Research*, 22, 2021.

Shenglong Zhou, Ouya Wang, Ziyan Luo, Yongxu Zhu, and Geoffrey Ye Li. Preconditioned inexact stochastic ADMM for deep models. *Nature Machine Intelligence*, 8(2):234–245, 2026. doi: 10.1038/s42256-026-01182-3.

A KKT multipliers as backpropagation adjoints

For a fixed set of weights θ and one sample (x, y), consider the activity-constrained inner problem induced by (1). At any feasible KKT point of this fixed-weight problem, the Lagrange multipliers satisfy the same reverse recursion as the backpropagation adjoints, and evaluating the weight derivative of the Lagrangian at that point gives the BP weight gradient. We recapitulate this point for the standard Lagrangian (the classical observation of LeCun [1988], recorded here in our notation conventions) and then show that the quadratic augmentation does not change the endpoint.

Throughout, the backpropagation adjoint at hidden layer i is the total derivative of the supervised loss after composing the downstream layers:

$$\delta_i \;:=\; \frac{d}{dh_i}\,\tfrac{1}{2}\big\|y - W_L h_{L-1}\big\|^2 \qquad \text{evaluated along the forward pass} \quad h_j = \sigma(W_j h_{j-1}),\; h_0 = x. \tag{15}$$

By the chain rule,

$$\delta_{L-1} = -W_L^\top\big(y - W_L h_{L-1}\big), \qquad \delta_i = W_{i+1}^\top \operatorname{diag}\big(\sigma'(W_{i+1}h_i)\big)\,\delta_{i+1} \quad \text{for } i = 1, \dots, L-2. \tag{16}$$

A.1 Standard Lagrangian

Setting $\rho = 0$ in (5) gives the standard Lagrangian

$$\mathcal{L}_0(h, \theta, \lambda) \;=\; \tfrac{1}{2}\big\|y - W_L h_{L-1}\big\|^2 \;+\; \sum_{i=1}^{L-1} \lambda_i^\top\big(h_i - \sigma(W_i h_{i-1})\big). \tag{17}$$

Proposition 1. *For fixed θ, at any feasible KKT point of the activity-constrained inner problem, the multipliers of $\mathcal{L}_0$ satisfy $\lambda_i = -\delta_i$ for $i = 1, \dots, L-1$. Consequently, the derivative $\nabla_{W_i}\mathcal{L}_0$ evaluated at this endpoint equals the backpropagation gradient of $\frac{1}{2}\|y - W_L h_{L-1}\|^2$ with respect to W_i.*

Proof. Primal feasibility forces $h_i = \sigma(W_i h_{i-1})$ for $i = 1, \dots, L-1$, so h lies on the forward pass.

Boundary, $i = L-1$: The activation h_{L-1} enters $\mathcal{L}_0$ via the supervised loss and the layer-$(L-1)$ residual:

$$\nabla_{h_{L-1}}\mathcal{L}_0 \;=\; -W_L^\top\big(y - W_L h_{L-1}\big) \;+\; \lambda_{L-1}. \tag{18}$$

Stationarity gives $\lambda_{L-1} = W_L^\top(y - W_L h_{L-1}) = -\delta_{L-1}$.

Recursion, $1 \le i \le L-2$: The activation h_i enters the layer-i residual as the LHS (multiplier λ_i) and the layer-$(i+1)$ residual through $\sigma(W_{i+1}h_i)$ (multiplier λ_{i+1}):

$$\nabla_{h_i}\mathcal{L}_0 \;=\; \lambda_i \;-\; W_{i+1}^\top \operatorname{diag}\big(\sigma'(W_{i+1}h_i)\big)\,\lambda_{i+1}. \tag{19}$$

Stationarity gives $\lambda_i = W_{i+1}^\top \operatorname{diag}(\sigma'(W_{i+1}h_i))\,\lambda_{i+1}$. Induction from the boundary yields $\lambda_i = -\delta_i$ for all i.

Weight gradient: For $i = 1, \dots, L-1$, only the i-th layer constraint involves W_i:

$$\nabla_{W_i}\mathcal{L}_0 \;=\; -\operatorname{diag}\big(\sigma'(W_i h_{i-1})\big)\,\lambda_i\,h_{i-1}^\top \;=\; \operatorname{diag}\big(\sigma'(W_i h_{i-1})\big)\,\delta_i\,h_{i-1}^\top, \tag{20}$$

which is the backpropagation gradient at layer i. For the readout, $\nabla_{W_L}\mathcal{L}_0 = -(y - W_L h_{L-1})\,h_{L-1}^\top$, the linear-regression gradient. □

The minus sign in $\lambda_i = -\delta_i$ is only a convention: it comes from writing the residual as $r_i = h_i - \sigma(W_i h_{i-1})$. With the opposite residual convention, the multipliers would have the opposite sign. The invariant statement is that the multiplier recursion carries exactly the reverse-mode adjoint information, and the induced weight derivative is the BP gradient.

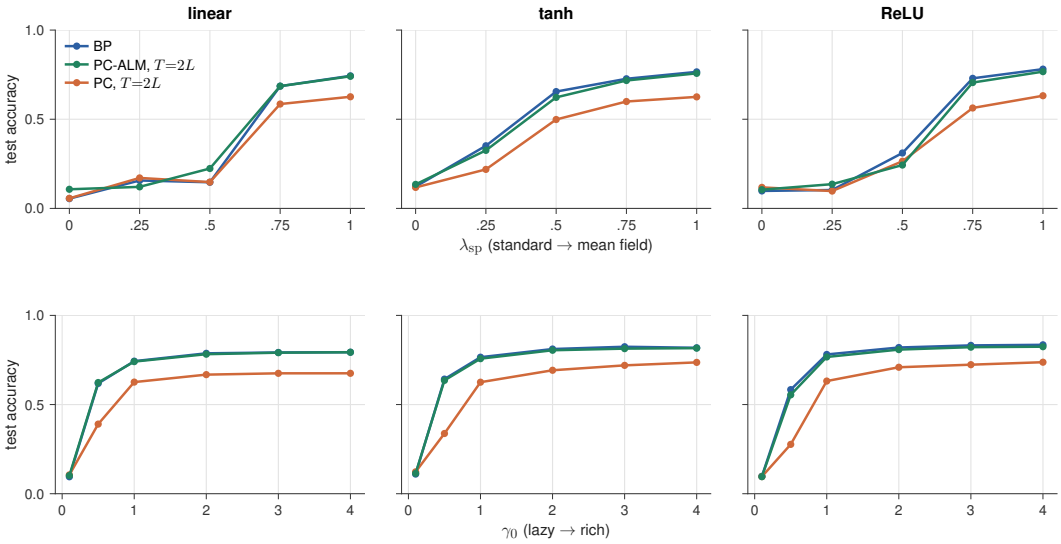


Figure 6: Parameterization sweep on Fashion-MNIST at fixed $(N, L) = (32, 64)$, single seed, $\alpha = 1$. Top row: λ_{sp} interpolates from standard parameterization (λ_{sp}=0) to mean field (λ_{sp}=1) at γ_0=1. Bottom row: γ_0 traces the lazy-to-rich axis within the mean-field family at λ_{sp}=1. PC-ALM at T=2L tracks BP across both axes; PC at T=2L tracks BP in the lazy regime but lags as the parameterization becomes rich.

A.2 Augmented Lagrangian

The augmented Lagrangian (5) adds the quadratic penalty $\frac{\rho}{2}\sum_{i=1}^{L-1}\|r_i\|^2$ to $\mathcal{L}_0$, where $r_i := h_i - \sigma(W_i h_{i-1})$.

Proposition 2. *For the fixed-weight activity problem, the KKT conditions and endpoint weight derivative of $\mathcal{L}_\rho$ coincide with those of $\mathcal{L}_0$. In particular, $\lambda_i = -\delta_i$ at any feasible KKT point, and $\nabla_{W_i}\mathcal{L}_\rho$ evaluated at that endpoint equals the BP weight gradient.*

Proof. The augmentation $\frac{\rho}{2}\sum_i \|r_i\|^2$ is a smooth function of r whose value and all h-, W-derivatives are linear in r and therefore vanish at $r = 0$. Primal feasibility forces $r = 0$, so the stationarity conditions and weight gradient of $\mathcal{L}_\rho$ at any feasible point reduce to those of $\mathcal{L}_0$, and Proposition 1 applies. □

Remark. Proposition 2 is endpoint-only: it characterizes any feasible KKT point but does not say the PC-ALM iteration reaches one. Linear-regime convergence to the KKT point is in Appendix C.

Feedforward graph extension. The same calculation applies to feedforward graphs with skips. At a hidden node, stationarity collects the multiplier on every incoming constraint and subtracts the Jacobian-transpose pullback from every outgoing constraint. On a DAG, solving these equations in reverse topological order gives the usual reverse-mode accumulation. The chain proof above is the single-parent, single-child specialization.

B Architectures

In our main equations we use the layer equations as $h_i = \sigma(W_i h_{i-1})$ for notational simplicity. The experiments use the scaled architectures of Innocenti et al. [2026], where each layer carries a scalar constant that initializes and stabilizes PC at depth and width. The architectures and constants are below.

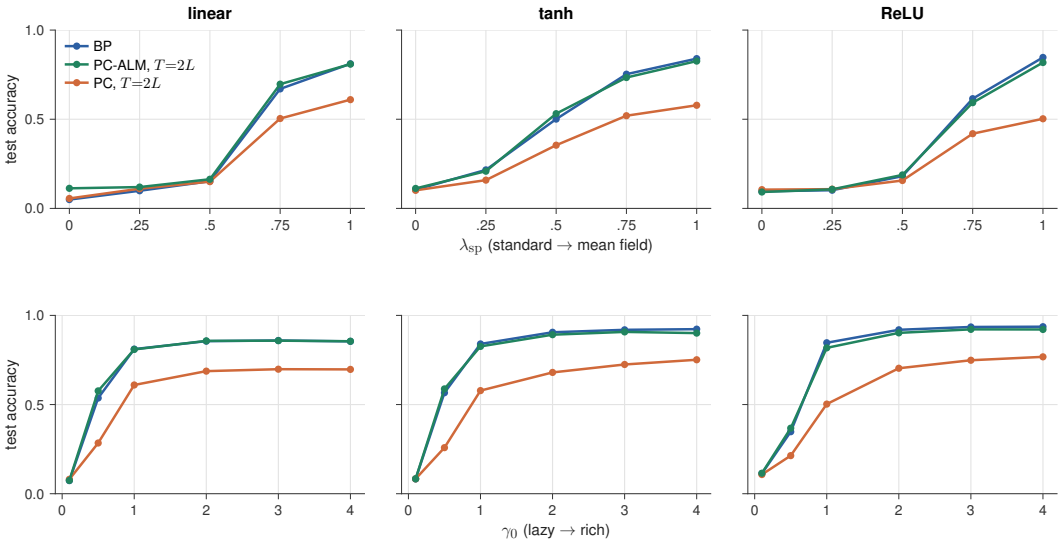


Figure 7: MNIST counterpart of Figure 6. Same axes, same fixed $(N, L) = (32, 64)$, same conclusion: PC-ALM tracks BP across the parameterization sweep; PC lags in the rich regime.

B.1 Architectures

A depth-L network has weights $W_1, \ldots, W_L$ and free hidden states $h_1, \ldots, h_{L-1}$ of width N, with $h_0 = x \in \mathbb{R}^D$ clamped and prediction $\hat{y} \in \mathbb{R}^C$. Each layer carries a scalar pre-multiplier a_i.

Chain MLP.

$$h_1 \;=\; a_1\, W_1 h_0, \qquad h_i \;=\; a_i\, W_i\, \sigma(h_{i-1}) \quad (2 \le i \le L-1), \qquad \hat{y} \;=\; a_L\, W_L\, \sigma(h_{L-1}). \tag{21}$$

The input layer is linear; interior and readout layers apply $\sigma \in \{\mathrm{id},\ \tanh,\ \mathrm{ReLU}\}$ before the linear map.

Residual MLP.

$$h_1 \;=\; a_1\, W_1 h_0, \qquad h_i \;=\; h_{i-1} + a_i\, W_i\, \sigma(h_{i-1}) \quad (2 \le i \le L-1), \qquad \hat{y} \;=\; a_L\, W_L\, \sigma(h_{L-1}). \tag{22}$$

Input and readout layers are unchanged from the chain; interior layers add an identity skip from the previous layer. The experiments use this topology.

B.2 Mean-field point

At the mean-field parameterization of Innocenti et al. [2026]—hidden activation exponent $a = 1/2$, init exponent $b = 0$, learning-rate exponent $c = 0$, output-scale exponent $d = 1/2$, residual depth exponent $\alpha_{\text{res}} = 1/2$ (distinct from the PC-ALM dual rate α)—the per-layer pre-multipliers are

$$a_1 \;=\; \frac{1}{\sqrt{D}}, \qquad a_i \;=\; \frac{1}{\sqrt{L\,N}} \quad (2 \le i \le L-1), \qquad a_L \;=\; \frac{1}{\gamma_0\, N}, \tag{23}$$

with all weights drawn $\mathcal{N}(0, 1)$ entrywise. The Adam learning rate is coupled to the parameterization [Innocenti et al., 2026],

$$\eta_{\text{Adam}} \;=\; \eta_0\, \gamma_0^2\, \sqrt{N/L}, \tag{24}$$

which collapses to the μPC default [Innocenti et al., 2025] at $\gamma_0 = 1$.

C Linear PC-ALM dynamics

This appendix isolates Algorithm 1 in the fixed-weight, linear-activation regime. With $\sigma = \mathrm{id}$ and weights frozen, one PC-ALM inner step is an affine map on the joint activity-dual state (h, λ). We

give the block iteration matrix M, prove convergence to the KKT point under the spectral-radius condition $\mathrm{spr}(M) < 1$, and recover the per-mode Jury bound as a corollary. Architecture-specific information (chain vs. skip MLP) enters only through the block constraint operator A; the explicit forms are in Appendix B.

C.1 Setup

Take the constrained problem (1) with $\sigma = \mathrm{id}$ and weights θ fixed. The free hidden states are $h = \{h_i\}_{i=1}^{L-1}$, and the multipliers are $\lambda = \{\lambda_i\}_{i=1}^{L-1}$. The residuals can be stacked as a column vector,

$$r(h) \;=\; \begin{bmatrix} r_1 \\ \vdots \\ r_{L-1} \end{bmatrix}, \qquad r_i \;=\; h_i - W_i\, h_{i-1}, \tag{25}$$

The residuals are affine in h and we write $r(h) = Ah + b$, where A is block lower-bidiagonal with I on the diagonal and the negative per-layer linear map below; b absorbs the clamped-input contribution from $h_0 = x$. For a plain chain,

$$A_{\text{chain}} \;=\; \begin{bmatrix} I & & & & \\ -W_2 & I & & & \\ & -W_3 & I & & \\ & & \ddots & \ddots & \\ & & & -W_{L-1} & I \end{bmatrix}, \qquad b_{\text{chain}} \;=\; \begin{bmatrix} -W_1 x \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{26}$$

The skip form has the same block pattern, with each sub-diagonal block replaced by the negative skip-layer prediction map; the scaled architecture forms are listed in Appendix B.

The supervised loss touches only h_{L-1}:

$$\ell(h) \;=\; \tfrac{1}{2}\,\big\|y - Ch\big\|^2, \qquad C \;=\; \big[\,0 \;\cdots\; 0 \;\; W_L\,\big], \qquad B \;:=\; C^\top C. \tag{27}$$

We keep this block separate from the constraint operator: multipliers attach to the network equations $Ah + b = 0$, not to the supervised residual $y - Ch$. The supervised term enters the activity curvature through B and therefore affects the full stability condition below. The augmented Lagrangian (5) in this regime is

$$\mathcal{L}_\rho(h, \lambda) \;=\; \tfrac{1}{2}\big\|y - Ch\big\|^2 \;+\; \lambda^\top\big(Ah + b\big) \;+\; \tfrac{\rho}{2}\,\big\|Ah + b\big\|^2. \tag{28}$$

C.2 One PC-ALM step

Algorithm 1 writes the inference loop as in-place assignments. For the dynamical-systems analysis here, t indexes the repeated inference step, and $h(t), \lambda(t)$ denote the full stacked hidden-state and multiplier vectors before step t. Unrolling one in-place primal-dual cycle gives

$$h(t+1) \;=\; h(t) \;-\; \eta_h\,\big[-C^\top(y - Ch(t)) \;+\; A^\top\lambda(t) \;+\; \rho\, A^\top(Ah(t) + b)\,\big], \tag{29}$$

$$\lambda(t+1) \;=\; \lambda(t) \;+\; \alpha\,\big(Ah(t+1) + b\big). \tag{30}$$

Both updates are affine in (h, λ), so PC-ALM is an affine dynamical system on the joint state.

C.3 Block iteration matrix

Suppose (29)–(30) has a fixed point $(h^\star, \lambda^\star)$. In error coordinates $E(t) := h(t) - h^\star$ and $R(t) := \lambda(t) - \lambda^\star$, the affine offsets cancel, leaving the linear recursion

$$\begin{bmatrix} E(t+1) \\ R(t+1) \end{bmatrix} \;=\; M \begin{bmatrix} E(t) \\ R(t) \end{bmatrix}, \qquad M \;=\; \begin{bmatrix} I - \eta_h(B + \rho\, A^\top A) & -\eta_h\, A^\top \\ \alpha\, A\big(I - \eta_h(B + \rho\, A^\top A)\big) & I - \alpha\eta_h\, AA^\top \end{bmatrix}. \tag{31}$$

Chain and skip MLPs share this expression; only A changes (Appendix B).

C.4 Main result: convergence to the KKT point and BP gradient

Proposition 3 (Linear convergence of PC-ALM). *Fix weights θ and take $\sigma = \mathrm{id}$, $\rho > 0$, $\alpha > 0$, $\eta_h > 0$. If $\mathrm{spr}(M) < 1$, then for any initialization $(h(0), \lambda(0))$ the iteration (29)–(30) converges linearly to a unique fixed point $(h^\star, \lambda^\star)$, and:*

1. *$h^\star$ satisfies $Ah^\star + b = 0$ (the linear forward pass);*
2. *$\lambda^\star$ are the negative BP adjoints of $\frac{1}{2}\|y - Ch\|^2$ at the linear forward pass;*
3. *the weight derivative at the fixed point, $\nabla_\theta \mathcal{L}_\rho(h^\star, \theta, \lambda^\star)$, equals the BP weight gradient.*

Proof. In affine coordinates the iteration takes the form $z(t+1) = Mz(t) + c$ for $z = (h, \lambda)$ and a constant c absorbing the input clamp b and supervised-loss offset. $\mathrm{spr}(M) < 1$ implies $I - M$ is invertible, so the fixed-point equation $z^\star = Mz^\star + c$ has a unique solution. In error coordinates $E(t), R(t)$ defined above, the affine offset cancels, leaving $z(t) - z^\star = M^t(z(0) - z^\star)$, which converges to 0 at rate $\mathrm{spr}(M)$.

At the fixed point, the dual update (30) forces $Ah^\star + b = 0$ since $\alpha > 0$. The activity stationarity (29) then reduces to $-C^\top(y - Ch^\star) + A^\top\lambda^\star = 0$. These are the KKT conditions of the linear constrained least-squares problem

$$\min_h \ \tfrac{1}{2}\big\|y - Ch\big\|^2 \quad \text{s.t.} \quad Ah + b \ = \ 0. \tag{32}$$

Appendix A (specialized to $\sigma = \mathrm{id}$, $r_i = h_i - W_i h_{i-1}$) identifies these multipliers as the negative BP adjoints and the weight derivative at the KKT point as the BP weight gradient. □

C.5 Modewise Jury bound

To get a clean per-mode picture, drop the supervised-loss block ($B = 0$); only the constraint penalty remains in M. With the SVD $A = U\Sigma V^\top$ and mode coordinates $x^{(i)}(t) := v_i^\top E(t)$, $y^{(i)}(t) := u_i^\top R(t)$, each positive singular mode evolves independently under the 2×2 map

$$M_i \ = \ \begin{bmatrix} 1 - \eta_h \rho \sigma_i^2 & -\eta_h \, \sigma_i \\ \alpha\sigma_i \left(1 - \eta_h \rho \sigma_i^2\right) & 1 - \alpha\eta_h \, \sigma_i^2 \end{bmatrix}, \tag{33}$$

where σ_i denotes the i-th singular value of A (not the activation σ, which is the identity throughout this appendix).

Corollary 1 (Schur stability per mode). *For $\eta_h, \rho, \alpha > 0$ and $\sigma_i > 0$, M_i is Schur-stable iff*

$$\eta_h \, \sigma_i^2 \, (2\rho + \alpha) \ < \ 4. \tag{34}$$

Across constraint modes, the bound binds at $\sigma_{\max}(A)$, which sets the activity-step ceiling for the constraint-only iteration.

The eigenvalues of M_i are roots of $\mu^2 - \mathrm{tr}(M_i)\,\mu + \det(M_i) = 0$ and form a complex conjugate pair when the discriminant $\mathrm{tr}(M_i)^2 - 4\det(M_i)$ is negative; Appendix C.6 works out the precise complex-eigenvalue regime and shows the per-mode response is then a damped oscillation rather than monotone decay. This corollary is not the full stability test when $B \neq 0$: the exact condition remains $\mathrm{spr}(M) < 1$, and readout/output scaling enters through B. In the special case where $B\,v_i = \nu_i v_i$ is diagonal in the same right singular basis as A, the same Jury calculation gives

$$\eta_h \left(2\nu_i + (2\rho + \alpha)\sigma_i^2\right) < 4. \tag{35}$$

Null modes ($\sigma_i = 0$) sit outside the constraint dynamics and are governed by any other curvature in the problem, namely the B block of M.

C.6 Per-mode radius and the role of α

The characteristic polynomial of M_i has trace and determinant

$$\mathrm{tr}(M_i) \ = \ 2 - \eta_h \, \sigma_i^2 \, (\rho + \alpha), \qquad \det(M_i) \ = \ 1 - \eta_h \, \rho \, \sigma_i^2. \tag{36}$$

Crucially, $\det(M_i)$ does not depend on α; only the trace does. In the complex-eigenvalue regime $\eta_h \sigma_i^2 (\rho + \alpha)^2 < 4\alpha$, the two roots are conjugate with $|\mu_\pm|^2 = \det(M_i)$, so

$$|\mu_\pm| \;=\; \sqrt{1 - \eta_h \, \rho \, \sigma_i^2}\,. \tag{37}$$

For fixed (η_h, ρ, σ_i), varying α slides the two eigenvalues along the circle of this radius centered at the origin in the complex plane; only the phase changes. Different singular values of A contribute different circles, so the per-mode complex eigenvalues of M populate an annulus

$$\sqrt{1 - \eta_h \, \rho \, \sigma_{\max}(A)^2} \;\le\; |\mu| \;\le\; 1 \tag{38}$$

in the complex plane. Two consequences:

1. The spectral radius of the complex-mode eigenvalues is set by η_h, ρ, and $\sigma_{\max}(A)$ alone. The dual rate α controls phase (oscillation frequency) but not the magnitude of decay. To accelerate contraction one must push $\eta_h \rho \sigma_{\max}^2$ closer to the Jury ceiling, not push α.
2. As α varies through the boundary $\eta_h \sigma_i^2 (\rho + \alpha)^2 = 4\alpha$, eigenvalues collide on the real axis and split into a complex conjugate pair (or vice versa). At $\alpha = 0$ the per-mode block is upper triangular with eigenvalues 1 and $1 - \eta_h \rho \sigma_i^2$ on the real axis; the 1-eigenvalue is the dual block's invariance.

Figure 4 (in the main text) shows this on a small skip MLP at fixed weights: eigenvalues of M populate the annulus the per-mode bound predicts, and the activity and dual traces show monotone descent at $\alpha = 0$ and damped oscillation at $\alpha > 0$.

C.7 Reduction to penalty PC at $\alpha = 0$

At $\alpha = 0$ the dual is pinned, $\lambda(t) \equiv \lambda(0)$, and M_i collapses to the upper-triangular block

$$M_i\big|_{\alpha=0} \;=\; \begin{bmatrix} 1 - \eta_h \rho \sigma_i^2 & -\eta_h \, \sigma_i \\ 0 & 1 \end{bmatrix}. \tag{39}$$

The dual mode is invariant (eigenvalue 1), so the joint activity–dual map is not Schur-stable. The activity block is gradient descent on the penalty $\frac{\rho}{2}\|Ah + b\|^2$ along v_i, with the standard penalty-PC stability bound $\eta_h \rho \sigma_i^2 < 2$. Taking the $\alpha \downarrow 0$ limit of Corollary 1 recovers the same activity-block inequality.

C.8 Chain vs. skip

The constraint part of the dynamics is written in A and $\sigma_i(A)$; chain and skip MLPs differ through this operator (Appendix B). The supervised readout contributes the separate block B.

Chain. The off-diagonal block of A is $-W_i$. The depth products $W_j W_{j-1} \cdots$ appear in the inverse/Green's function of this lower-bidiagonal system, so depth mainly shows up through the spectrum and conditioning of A. Large singular values tighten the activity-step ceiling; small singular values create slow residual modes. Both effects contribute to the finite-T inference difficulty for deep linear PC.

Skip. The off-diagonal block is $-(I + sW_i)$ with $s = 1/\sqrt{LN}$. This makes A_{skip} a discrete difference operator plus a small random perturbation. Under bounded weights, $\sigma_{\max}(A_{\text{skip}})$ remains controlled, so the Jury ceiling on η_h need not shrink just because L grows. Depth can still appear through small modes and conditioning, but the residual parameterization removes the most direct step-size collapse.

D PC-ALM credit-wave propagation

In PC-ALM, each hidden-layer weight update uses the composite credit

$$e_i^t \;=\; \lambda_i^t \;+\; \rho \, r_i^t, \tag{40}$$

whereas PC uses only ρr_i^t. At the feasible KKT point the residual vanishes and $e_i^t \to \lambda_i^t$, the negative BP adjoint (Appendix A). The finite-T question is how quickly e_i^t acquires useful magnitude and BP alignment at layers far from the readout.

Mode dynamics. In the fixed-weight linear regime of Appendix C with the supervised block B dropped, set $q := \eta_h \sigma^2$ for one singular mode of the constraint operator A. The PC-ALM mode polynomial is

$$\mu^2 \;-\; \big[\,2 - q(\rho+\alpha)\,\big]\,\mu \;+\; (1-\rho q) \;=\; 0, \tag{41}$$

stable when $q(2\rho+\alpha) < 4$ (Corollary 1). In the complex-eigenvalue regime the two roots are $\mu_\pm = r\, e^{\pm i\theta}$, with

$$r \;=\; \sqrt{1-\rho\,\eta_h\,\sigma^2}\,, \qquad \cos\theta \;=\; \frac{2-\eta_h\,\sigma^2(\rho+\alpha)}{2\sqrt{1-\rho\,\eta_h\,\sigma^2}}. \tag{42}$$

The magnitude depends on (ρ, η_h, σ) alone; α enters only through the phase (Appendix C.6). For low-frequency modes, $q \ll 1$ and

$$r \;\approx\; \exp\!\big(-\tfrac{1}{2}\,\rho\,\eta_h\,\sigma^2\big)\,, \qquad \theta \;\approx\; \sqrt{\alpha\,\eta_h}\;\sigma. \tag{43}$$

Skip-MLP dispersion. Under residual mean-field scaling, A is a layerwise first-difference operator plus a controlled random perturbation (Appendix B). Its long-wavelength singular values follow $\sigma(k) \approx 2\sin(k/2) \approx k$, with k the layer frequency. Substituting into (43) gives the low-k dispersion relation

$$\mu_\pm(k) \;\approx\; \exp\!\big(-\tfrac{1}{2}\,\rho\,\eta_h\,k^2 \;\pm\; i\,\sqrt{\alpha\,\eta_h}\,k\big)\,. \tag{44}$$

The real exponent is diffusive damping; the imaginary exponent is the phase of a damped wave with group velocity

$$v_{\text{credit}} \;=\; \sqrt{\alpha\,\eta_h} \qquad \text{layers per inference step.} \tag{45}$$

PC has no phase term: $\mu_{\text{PC}}(k) \approx \exp(-\rho\,\eta_h\,k^2)$ is pure heat diffusion. After T inner steps, an output-side credit impulse therefore reaches

$$R_{\text{PC}}(T) \;=\; O\Big(\sqrt{\rho\,\eta_h\,T}\Big)\,, \qquad R_{\text{ALM}}(T) \;\approx\; T\sqrt{\alpha\,\eta_h}\,, \tag{46}$$

the latter up to a diffusive front width $O\big(\sqrt{\rho\,\eta_h\,T}\big)$. The strict layerwise "light cone" is one layer per inner step in both cases; (46) measures the depth at which the local credit has crossed a usable amplitude and BP alignment, not where it first arrives.

Predicted inflection step. Equating $R_{\text{ALM}}(T)$ with the network depth L gives the inflection observed in Section 5.6,

$$t_{\text{infl}} \;\approx\; \frac{L}{\sqrt{\alpha\,\eta_h}}, \qquad \alpha_{\text{reach}} \;\approx\; \frac{L^2}{\eta_h\,T^2}. \tag{47}$$

At the conditioning step $\eta_h = 1/\lambda_{\max}$ with $\lambda_{\max} = \sigma^2_{\max}(A) \approx 4$ under residual mean-field, $\alpha_{\text{reach}} \approx 4(L/T)^2$. For $L = 64$, $T = 2L$ this gives $\alpha_{\text{reach}} \approx 1$, consistent with the inflection observed near $t \approx 2L$ at $\alpha = 1$ in the headline cell. The Jury bound (34) caps $\alpha < 4/(\eta_h\lambda_{\max}) - 2\rho$; at $\eta_h = 1/\lambda_{\max}$, $\rho = 1$ this is $\alpha < 2$, so the fastest stable reach in T inner steps is $T\sqrt{2/\lambda_{\max}}$. Cells with L exceeding this remain budget-limited even at the cap.

E Leaky multipliers

In PC-ALM, each multiplier acts as a perfect integrator of its residual and is reset between samples. A more biologically plausible mechanism is leaky integration, in which the multiplier decays to zero in the absence of input. This can be implemented by adding a leak κ on the multiplier update:

$$\lambda_i \;\leftarrow\; (1-\kappa)\,\lambda_i \;+\; \alpha\, r_i, \qquad 0 \le \kappa \le 1. \tag{48}$$

While more biologically plausible, leaky multiplier updates also arise naturally in distributed primal-dual methods [Koshal et al., 2011, Tang et al., 2018], where the leak regularizes the dual variables.

For $\kappa > 0$, a fixed point satisfies $\lambda_i = (\alpha/\kappa)\, r_i$. The activity and weight gradients (11) and (13) then equal those of PC with ρ replaced by $\rho + \alpha/\kappa$. A leaky equilibrium is therefore a PC equilibrium with a stiffer penalty. PC-ALM is the limit $\kappa \to 0$, and PC is the limit $\alpha/\kappa \to 0$. As discussed in Section 2, at convergence, PC-ALM gives up the so-called prospective configuration benefit of PC [Song et al., 2024]: in PC, the activations equilibrate at values that already move the output

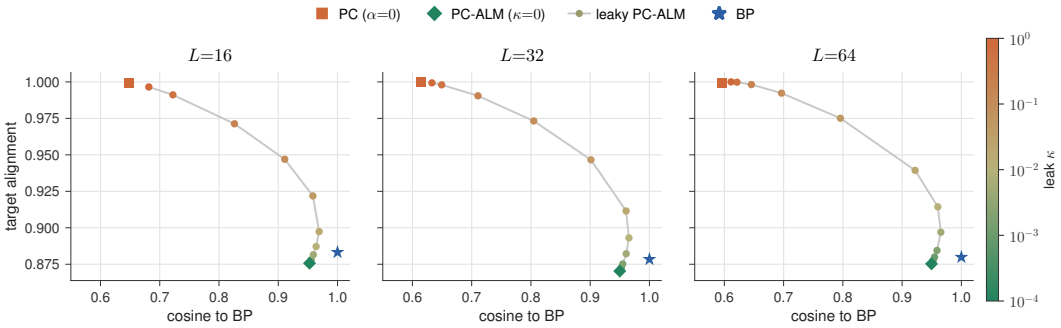


Figure 8: Target alignment against cosine to the BP gradient for leaky PC-ALM (48) with $\kappa \in [10^{-4}, 1]$ (color), PC, PC-ALM, and BP. ReLU residual MLPs with $N = 32$, $L \in \{16, 32, 64\}$ on Fashion-MNIST at initialization, batch size 1, $T = 4L$, $\alpha = 1$, $\eta_h = 0.75/\lambda_{\max}$. Mean over 27 paired draws of network and sample.

toward the target, making the weight update more similar to a backward Euler step [Alonso et al., 2024]. This admits sample efficiency benefits and alleviates interference in multi-task settings [Song et al., 2024, Oliviers et al., 2026]. We measure prospective configuration by output target alignment [Song et al., 2024, Oliviers et al., 2026], the cosine between the output error $y - \hat{y}$ and the first-order change in $\hat{y}$ produced by the weight update. Adding a leak can be viewed as interpolating between the prospective configuration benefits of PC and the credit propagation and BP alignment benefits of PC-ALM. We investigate this trade-off in Figure 8. Notably, the Pareto front bows toward the upper right, so intermediate leaks recover much of PC's target alignment while keeping most of PC-ALM's BP alignment.

F Complexity comparison

Two routes to BP-aligned gradients. Under the mean-field parameterization of Innocenti et al. [2026], the equilibrated PC energy rescaling satisfies $s(\theta) = 1 + \Theta(L/N)$ (Cor. 4.2; Eq. 19), so the architectural PC–BP mismatch scales as $\Theta(L/N)$ at activity equilibrium. Two routes therefore reduce the PC–BP mismatch: widening the network so that L/N is small, or running PC-ALM at fixed width. These cost, respectively, a width budget tied to depth and an adaptive auxiliary state.

A simplifying assumption on PC's inner budget. We compare the two routes at a target tolerance ε on the gradient mismatch. PC's gradient mismatch has two sources: an architectural mismatch $s(\theta) - 1 = \Theta(L/N)$ that persists even at perfect activity equilibrium, and a finite-inference mismatch from a finite inference budget and incomplete equilibration. We assume PC runs at a fixed activity budget $T_{\mathrm{PC}} = \Theta(1)$, independent of ε, large enough that the finite-inference mismatch is negligible beside the architectural term, so PC's only ε-dependence is its width N_{w}. We make no matching assumption for PC-ALM, whose inner count remains ε-dependent; the accounting thus favors PC.

Resource comparison. Per training sample, PC at width N_{w} runs T_{PC} activity-update steps (fixed, by the assumption above); PC-ALM at fixed width N_0 runs K_ε joint primal–dual iterations, in the linear fixed-weight setting of Proposition 3, to drive the mismatch below ε. The PC-ALM convergence rate used below is the linear fixed-weight rate of Proposition 3; the nonlinear ReLU experiment in Figure 9 is an empirical diagnostic rather than a consequence of this asymptotic argument.

The PC-ALM inner count follows from linear convergence. By Proposition 3, the joint primal–dual iteration contracts at rate $\mathrm{spr}(M) < 1$, where $\mathrm{spr}(\cdot)$ denotes the spectral radius. Driving the gradient mismatch below ε therefore requires

$$K_\varepsilon = O\left(\frac{\log(\varepsilon)}{\log\left(\mathrm{spr}(M)\right)}\right) \tag{49}$$

joint primal–dual iterations. For PC there is no auxiliary state, and by the fixed-budget assumption no ε-dependent inner count. Its gradient mismatch is set by the architectural rescaling $s(\theta) - 1 =$

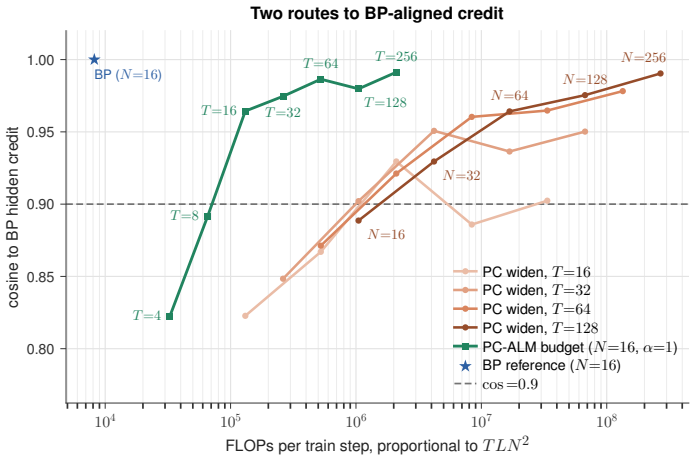


Figure 9: Two routes to BP-aligned hidden credit at $L = 32$, ReLU, Fashion-MNIST. Here T denotes the plotted per-sample update budget: activity steps for PC and joint primal-dual cycles for PC-ALM. Orange: PC at fixed inference budget T, sweeping hidden width N. Green: PC-ALM ($\alpha = 1$) at fixed $N = 16$, sweeping T. Blue star: BP reference at $N = 16$. The horizontal axis is per-step compute, proportional to TLN^2. In this $L = 32$, ReLU/Fashion-MNIST setting, the PC-ALM branch reaches 0.9 cosine similarity to BP at roughly an order of magnitude less compute than the PC width sweeps shown.

	Parameters	Memory (activations + multipliers)	Total time
PC-ALM (fixed width N_0)	LN_0^2	$2LN_0B$	$K_\varepsilon\, LN_0^2B$
PC at width N_w	LN_w^2	$LN_\mathrm{w}B$	$T_\mathrm{PC}\, LN_\mathrm{w}^2B$
Ratio (ALM / PC)	N_0^2/N_w^2	$2N_0/N_\mathrm{w}$	$K_\varepsilon N_0^2/(T_\mathrm{PC}N_\mathrm{w}^2)$

Table 1: Resource costs for PC-ALM at fixed width N_0 and PC at width N_w, both under mean-field scaling at matched gradient tolerance ε, with PC's inner budget T_PC held fixed and ε-independent.

$\Theta(L/N)$ [Innocenti et al., 2026, Eq. 19], so ε is controlled by the choice of width, $N_\mathrm{w}(\varepsilon) = \Theta(L/\varepsilon)$. Widening pays for tighter ε linearly in width, while PC-ALM pays for tighter ε logarithmically in joint iterations. The dominant costs are summarized in Table 1. Symbol guide: L is depth, B is batch size, N_0 is PC-ALM's fixed width, N_w is PC's width, K_ε is the number of joint PC-ALM primal–dual iterations to reach tolerance ε, T_PC is PC's fixed (ε-independent) activity-inference budget, λ denotes the ALM multiplier state, and M is the linear PC-ALM iteration matrix from Proposition 3.

Space asymptotics. Substituting $N_\mathrm{w} = \Theta(L/\varepsilon)$ into the bottom row of Table 1 gives the asymptotic ratios. Parameters scale as $\Theta(N_0^2\varepsilon^2/L^2)$ and activation memory as $\Theta(N_0\varepsilon/L)$. Both are pure architectural ratios, free of the inner-iteration counts; PC-ALM dominates PC on space whenever $N_0\varepsilon \ll L$.

Time asymptotics. Substituting $N_\mathrm{w} = \Theta(L/\varepsilon)$ and K_ε asymptotics gives

$$\frac{\mathrm{time}_\mathrm{ALM}}{\mathrm{time}_\mathrm{PC}} = O\left(\frac{N_0^2\varepsilon^2}{L^2}\cdot\frac{\log(\varepsilon)}{T_\mathrm{PC}\,\log\left(\mathrm{spr}(M)\right)}\right), \tag{50}$$

for fixed L, N_0 and a stable PC-ALM iteration matrix M with $\mathrm{spr}(M) < 1$. Under the fixed-budget assumption $T_\mathrm{PC} = \Theta(1)$, the ratio is $O(|\varepsilon^2\log(\varepsilon)|)$. The architectural factor $N_0^2\varepsilon^2/L^2$ drives the entire ratio to zero as ε tightens, provided the PC-ALM spectral gap is bounded away from zero, since $\varepsilon^2\log(1/\varepsilon) \to 0$. PC-ALM therefore dominates PC on time in the small-ε limit. If $\mathrm{spr}(M) \to 1$ with depth, weights, or conditioning, the constant can degrade. A symmetric treatment of PC's inner budget, charging PC an ε-dependent inference budget as well, would cancel the $\log(1/\varepsilon)$ factor up to the differing conditioning of the two inference problems.

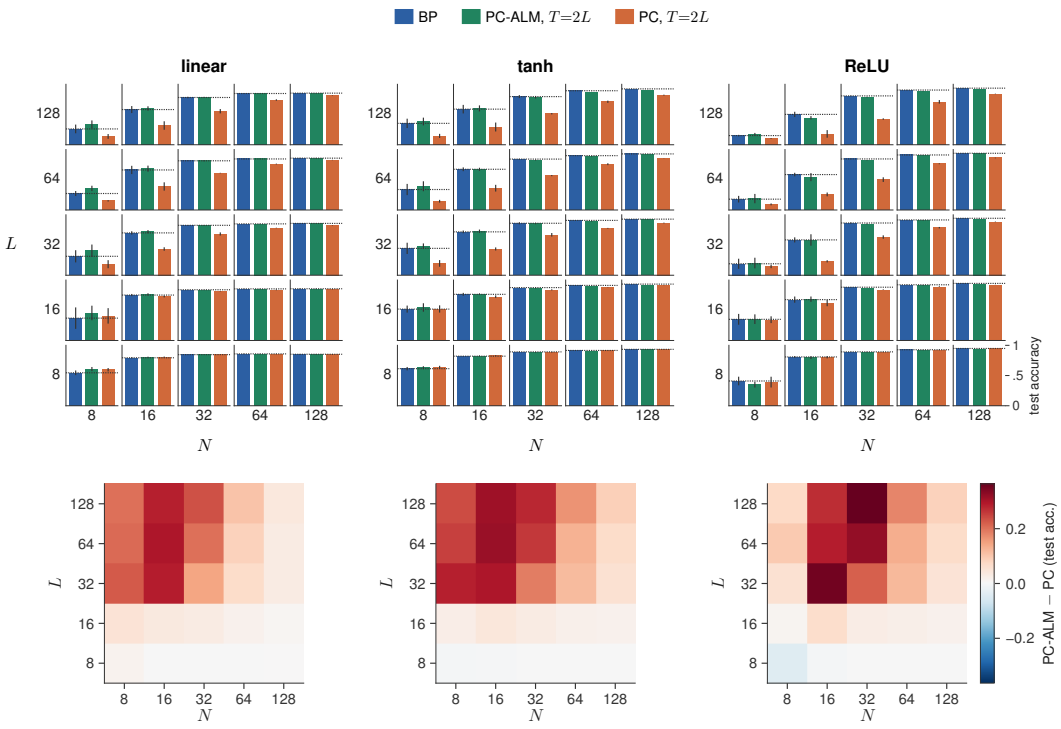


Figure 10: MNIST counterpart of Figure 2. Same (N, L) grid, three seeds, $\alpha = 1$. Same regime structure: PC-ALM at $T = 2L$ tracks BP across the grid; PC drops in deep narrow networks; the gain concentrates in deep, narrow cells.

G Experimental details

For our experiments we closely follow the setup of Innocenti et al. [2026], using their residual MLP architecture (Appendix B) and mean-field parameterization. The parameterization stabilizes $\sigma_{\max}(A)$, so a default η_h would suffice in principle. However, $\sigma^2_{\max}(A)$ is sensitive in shallow networks, and α and η_h are coupled. We therefore compute an empirical $\lambda_{\max} := \sigma^2_{\max}(A)$ for each (N, L) and dataset, and set $\eta_h = 1/\lambda_{\max}$. This places α in an interpretable $[0, 2]$ range. All networks are trained for one epoch on MNIST [LeCun et al., 1998] and Fashion-MNIST [Xiao et al., 2017]. We additionally swept η_h for several PC networks on a tight grid, and observed negligibly improved performance at higher η_h followed by collapse beyond $2/\lambda_{\max}$ as expected. The η_h sweep was unable to improve PC's performance in any meaningful way compared to PC-ALM, which we use as justification for the default η_h choice. We use a batch size of 64.

G.1 Convolutional networks on CIFAR-10 and Tiny ImageNet

We train ResNet-18 [He et al., 2016] on CIFAR-10 [Krizhevsky, 2009] and Tiny ImageNet [Le and Yang, 2015] with BP, PC, and PC-ALM (Table 2). Each convolution is followed by GroupNorm with a single group [Wu and He, 2018]. We do not use BatchNorm, since it couples the examples in a minibatch. All runs use LeakyReLU, a squared-error loss on one-hot targets, batch size 128, 50 epochs, and AdamW with weight decay 5×10^{-4} and a warmup-cosine schedule. PC and PC-ALM use $T = 96$ inference steps.

With these settings, PC-ALM improves on PC by 18.3 points on CIFAR-10 and by 7.7 Top-1 points on Tiny ImageNet, and remains below BP on both.

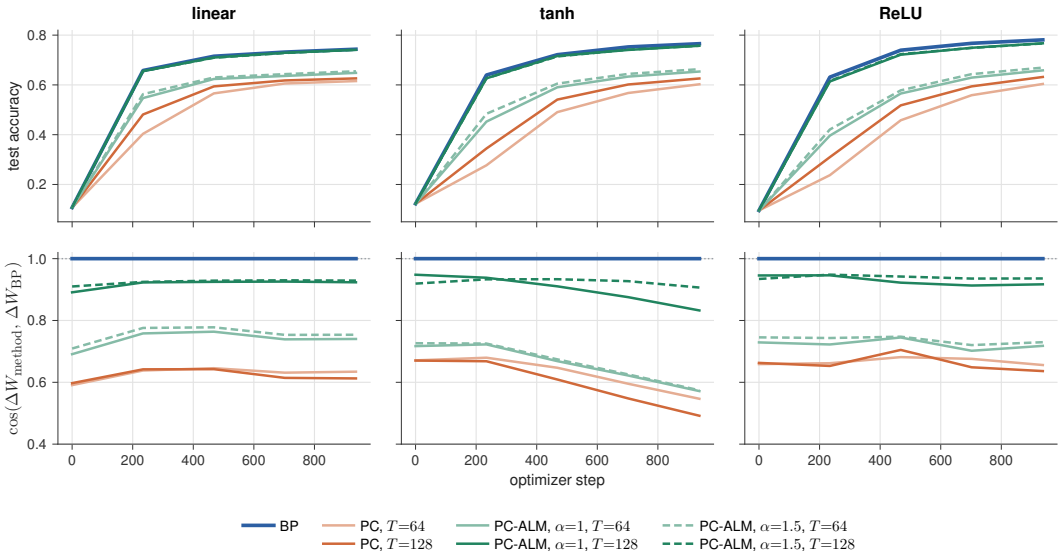


Figure 11: Training curves for one epoch at $N = 32$, $L = 64$ on Fashion-MNIST across linear / tanh / ReLU.

Table 2: ResNet-18 test accuracy (%), mean $\pm$ s.d. over seeds. CIFAR-10: mean over the last five of 50 epochs, three seeds. Tiny ImageNet: validation accuracy at epoch 50, five seeds.

Dataset	BP	PC	PC-ALM
CIFAR-10	90.44 ± 0.28	55.93 ± 1.85	74.18 ± 0.71
Tiny ImageNet, Top-1	44.93 ± 0.19	12.00 ± 0.95	19.72 ± 0.33
Tiny ImageNet, Top-5	64.78 ± 0.26	31.00 ± 1.41	43.36 ± 0.31

H Displaced activity initialization

PC performance depends on the initialization of the activities [Pinchetti et al., 2026]. Typically, in PC, activities are initialized to forward-pass values, generally motivated by the fact that this activity pattern is close to inference equilibrium. In PC-ALM, a forward initialization admits an additional justification in that the forward-pass values are the activations at the KKT point. An issue shared by both PC and PC-ALM is how performance degrades as initialization deviates from equilibrium. Here, we introduce a displaced initialization that allows us to test this degradation. We test on Fashion-MNIST with a residual MLP, $(N, L) = (32, 64)$, ReLU, and $T = 2L$. Before inference, each activation is set to:

$$h_i(0) \;=\; h_i^{\text{fwd}} \;+\; c\, s_i\, \xi_i, \qquad \xi_i \sim \mathcal{N}(0, I), \tag{51}$$

where h_i^{fwd} is the forward-pass value and s_i is its root-mean-square. In PC-ALM, we initialize the multipliers to 0. Setting $c = 0$ recovers the initialization of the main experiments, and at $c = 1$ the noise is as large as the activity. PC and PC-ALM receive the same draws. Table 3 reports test accuracy after one epoch.

Table 3: Test accuracy (%) under the displaced initialization (51), mean $\pm$ s.d. over five seeds. Fashion-MNIST, $(N, L) = (32, 64)$, ReLU, $T = 2L$, one epoch. BP reaches 76.2.

c	0	0.03	0.1	0.3	1	3
PC	61.7 ± 2.2	57.3 ± 3.6	54.6 ± 4.4	49.9 ± 5.1	39.5 ± 5.2	25.6 ± 8.1
PC-ALM	75.2 ± 0.5	72.5 ± 1.2	67.3 ± 1.3	56.8 ± 2.8	24.1 ± 5.1	12.8 ± 3.9

In particular, we observe that both methods degrade as c is increased. Notably, we see PC-ALM outperform PC for c up to 0.3. But at $c \geq 1$, PC beats PC-ALM, though in a regime where both networks are performing poorly on the task (test accuracy below 40%).

Intuitively, this flip occurs because in the high-displaced regime, PC-ALM's multipliers integrate large residuals that are unrelated to the output loss. This emphasizes that both PC and PC-ALM are sensitive to initialization, and further that PC-ALM can fall behind PC when initialized far from the KKT point.

I Limitations

Structural restrictions. The convergence proof for PC-ALM (Appendix C, Proposition 3) and the credit-wave analysis of Appendix D are restricted to networks with identity activations and no bias terms. Our experiments use activations (identity, tanh, ReLU) across the full $(N, L) \in \{8, 16, 32, 64, 128\}^2$ grid and PC-ALM closes the PC-BP gap throughout, but a convergence guarantee covering this setting is open. This simplification to linear and bias-free networks mirrors the broader PC literature: the saddle analysis of Innocenti et al. [2024] likewise uses linear networks without biases for theoretical development, and nonlinear networks for empirical verification. The μPC scaling analysis of Innocenti et al. [2025] is likewise linear and bias-free in its theoretical content while running nonlinear (bias-free) experiments.

Empirical scope. Experiments are restricted to MNIST, Fashion-MNIST, CIFAR-10, and Tiny ImageNet, with residual MLPs (one epoch of training) and ResNet-18. Broader evaluation on larger and more diverse datasets, longer training horizons, and architectures beyond residual MLPs and ResNet-18 (attention-based) remains future work. The width-depth grid probes the regime relevant to the PC-BP gap but is small relative to production networks.

Inference budgets. The correspondence between the weight update and the BP gradient (Appendix A) only holds at equilibrium. Such an equilibrium may fail to be reached with a finite inference budget, causing gradient misalignment. Moreover, the inference budget necessary to reach equilibrium (within a tolerance) grows with network depth. Far from the KKT point, for example under a displaced activity initialization, PC-ALM can fall behind PC (Section 5.8).

J Additional Related Work

There is a large body of work on biologically plausible, local learning methods. Target propagation [Bengio, 2014] and difference target propagation [Lee et al., 2015] have been shown to be related to predictive coding [Millidge et al., 2022b]. We believe that connections of target propagation to PC-ALM in particular merit investigation. The predictive coding literature often leads with a probabilistic framework [Rao and Ballard, 1999, Friston and Kiebel, 2009, Millidge et al., 2021, Salvatori et al., 2026]; the framing of PC as a quadratic penalty on the constrained optimization problem is, to our knowledge, less common (though see Seely [2025]). One benefit of the PC formalism is that it can handle non-DAG topologies as studied in Salvatori et al. [2022]. The AL framework also extends to non-DAG architectures; at least in a linear PCN, recurrent topologies change the KKT point but do not affect convergence results. With nonlinear activations, the primal-dual dynamics can support nonlinear systems with limit cycles and other nonlinear behavior, which is not possible in standard PC's gradient-flow inference; we leave this to future investigation. PC has also been cast in terms of sheaf theory, which offers tools for analyzing recurrent topologies [Seely, 2025]; closely related is the work of Bosca and Ghrist [2026], which extends the sheaf-theoretic framing to ReLU MLPs with resulting PC-like inference and learning steps. From this sheaf-theoretic perspective, the primal-dual dynamics of PC-ALM can be viewed as an application of the homological programming methods of Hansen and Ghrist [2019]. Prior work on "PC approximates backprop" is highlighted by Innocenti et al. [2026]; earlier work showed that with a "fixed prediction assumption" (a modification of standard PC), error propagation can converge to BP [Millidge et al., 2022c], albeit in a large number of iterations; this work was further studied in Rosenbaum [2022]. Finally, we remind the reader that while PC and PC-ALM address one aspect of BP's bio-implausibility, they do not address weight transport (the gradient of the energy still uses the transpose of the layer's weights [Bredenberg et al., 2023]).

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and Introduction state the paper's three contributions and their scope: (i) PC-ALM as finite-inference dual ascent on the augmented Lagrangian of the constrained-network problem, with the algorithm given in the Method section; (ii) exact BP gradients at equilibrium for linear PCNs, proven in the KKT-proof and Linear PC-ALM dynamics appendices; (iii) empirical evidence that PC-ALM matches BP at matched inference budget across $(N, L) \in \{8, 16, 32, 64, 128\}^2$ on MNIST and Fashion-MNIST, reported in the Results section; the main empirical claims concern residual MLPs on these two datasets, and Appendix G.1 reports ResNet-18 on CIFAR-10 and Tiny ImageNet, where PC-ALM improves on PC but remains below BP.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in the Limitations appendix (Appendix I) and the Results section: the exact-BP equivalence is established only for linear PCNs (Linear PC-ALM dynamics appendix); nonlinear results rely on a finite-T inference budget rather than convergence to equilibrium; PC-ALM can fall behind PC when initialized far from the KKT point (Section 5.8, Appendix H); PC-ALM doubles PC's per-layer activation memory; the main experiments cover residual MLPs on MNIST and Fashion-MNIST, convolutional results are limited to ResNet-18 on CIFAR-10 and Tiny ImageNet (Appendix G.1), and we do not test recurrent or transformer architectures.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.

- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The exact-BP-at-equilibrium result for linear PC-ALM is stated in the Results section with assumptions (linear activation, fixed weights, spectral-radius condition $\mathrm{spr}(M) < 1$ on the block iteration matrix); the KKT-proof appendix shows that at any feasible KKT point of the constrained-network problem the multipliers are the negative BP adjoints (general feedforward), and the Linear PC-ALM dynamics appendix proves convergence to that KKT point in the linear regime; cited prior results (LeCun [1988], Hestenes [1969]) are referenced where used.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Algorithm 1 in the Method section gives the full PC-ALM update; the Experimental details appendix specifies the residual MLP architecture (Architectures appendix), the mean-field parameterization of Innocenti et al. [2026], the $\eta_h = 1/\lambda_{\max}$ rule, the $\alpha = 1$ default, the $T \in \{L, 2L\}$ inference budgets, the $(N, L) \in \{8, 16, 32, 64, 128\}^2$ grid, three seeds for headline figures, and one epoch on MNIST and Fashion-MNIST; baselines (PC, BP) use the same architecture, parameterization, optimizer, and data pipeline; the ResNet-18 setup is given in Appendix G.1; code is released (see the next question).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same

dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code is released at `https://github.com/SakanaAI/pc-alm` under the MIT license, with instructions to reproduce the residual MLP width/depth grid on MNIST and Fashion-MNIST for BP, PC, and PC-ALM. MNIST and Fashion-MNIST are publicly distributed through standard library implementations.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

Justification: The Results section states the data, (N, L) grid, inference budget, activation set, and seed count; the Experimental details appendix gives the architecture, mean-field parameterization, the λ_{sp} and γ_0 parameterization sweeps, the per-network $\eta_h = 1/\lambda_{\max}$ choice, the α and ρ defaults, the optimizer, the batch size, and the training length; hyperparameter selection (in particular the η_h sweep used to defend the default) is described in the same appendix; the ResNet-18 settings are given in Appendix G.1.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: The headline (N, L) grid figures on Fashion-MNIST and MNIST report three independent seeds; whiskers show ± 1 sample standard deviation across seeds (`numpy.std` with `ddof=1`), where the factor of variability is random initialization and minibatch ordering at fixed hyperparameters; sweep-style figures (parameterization sweep, budget-to-BP) use a single seed and are labeled accordingly in their captions; the displaced-initialization table (five seeds) and the ResNet-18 table (three and five seeds) report mean $\pm$ s.d., and the leaky-multiplier figure reports means over 27 paired draws of network and sample.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: All training was performed on a single internal GPU cluster using NVIDIA H100 GPUs. A single (N, L) training run completes in under two minutes (mean ≈ 30 s); the headline grid (PC, PC-ALM, BP across the $(N, L) \in \{8, 16, 32, 64, 128\}^2$ cells, three seeds, two datasets, and three activations) totals approximately 25 GPU-hours. The full research

project, including discarded sweeps and parameterization studies that are not reported in the paper, used roughly 3× this budget. The ResNet-18 experiments (Appendix G.1) used about 100 H100 GPU-hours, most of it for Tiny ImageNet (about 20 GPU-hours per seed for BP, PC, and PC-ALM together).

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: We have read the NeurIPS Code of Ethics and the work conforms to it: the paper uses only publicly available image-classification benchmarks (MNIST, Fashion-MNIST, CIFAR-10, Tiny ImageNet), involves no human subjects or crowdsourcing, and introduces no models with foreseeable misuse risk.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [N/A]

Justification: This paper is foundational research on local credit-assignment dynamics for training feedforward networks; there is no direct path from PC-ALM to a specific positive or negative societal application beyond that of generic supervised neural-network training, which the NeurIPS guidelines indicate does not require a dedicated discussion.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper releases no pretrained generative models, no scraped datasets, and no artifacts with a high risk of misuse.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We use the publicly distributed MNIST and Fashion-MNIST datasets through standard library implementations and credit their original creators in the Experimental details appendix; CIFAR-10 and Tiny ImageNet are cited in Appendix G.1; the residual MLP architecture and mean-field parameterization follow Innocenti et al. [2026] and are cited where used.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The only new asset is the released code (https://github.com/SakanaAI/pc-alm), which is documented in its README and released under the MIT license; the paper introduces no new datasets or pretrained models.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core methodology (PC-ALM and the corresponding theoretical analysis) does not involve LLMs as an important, original, or non-standard component; LLMs were used only as a writing and editing aid.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.