UNDERSTANDING HOW INFERENCE SHAPES WEIGHT UPDATES IN PREDICTIVE CODING: FROM LABEL NOISE TO THE STATE-TO-WEIGHT INTERFACE

Anonymous authors
Paper under double-blind review

ABSTRACT

Predictive coding networks infer neural activities through iterative energy minimization and learn network parameters through local synaptic updates. In this work, we find that, under symmetric label noise, PC maintains its fit to clean samples while memorizing fewer incorrect labels than backpropagation. By analyzing state inference and weight updates separately, we find that inference absorbs an incorrect label into the hidden states rather than passing it on to the weights. For the same image, an incorrect label induces a larger feedforward prediction error than the true label, yet the residuals after inference are similar, yielding a higher state absorption ratio. Our theoretical analysis characterizes this separation, showing how inference absorbs prediction errors through state changes and how the resulting states shape both the residuals and parameter gradients used for weight updates. At fixed weights and inference states, using inferred inputs attenuates the weight-gradient response to label changes and alters its direction; replacing these inputs with feedforward states leads to greater noise memorization during training. PC also memorizes more class-structured incorrect labels than those from symmetric corruption. Together, these results show how inference filters target-induced changes before local updates accumulate in shared weights.

1 INTRODUCTION

Predictive coding (PC) learns in two stages. Inference adjusts neural activities to the current input and target; local weight updates change the predictions the network will make on later inputs (Rao & Ballard, 1999; Friston, 2010; Bogacz, 2017; Innocenti et al., 2026b). Analyses relating PC to backpropagation have examined when these local updates reproduce a global gradient (Whittington & Bogacz, 2017; Song et al., 2020; Millidge et al., 2020; Salvatori et al., 2021b). A network can fit a target during inference without learning to reproduce it in a forward pass. We ask how the result of inference becomes a weight update.

Label noise exposes this question. Studies of noisy-label learning describe how networks first fit shared structure and then acquire incorrect labels (Arpit et al., 2017; Toneva et al., 2019; Liu et al., 2020). In our experiments, PC fits clean examples while memorizing fewer incorrect labels than backpropagation (BP). BP increasingly fits those labels after clean fitting is established, and its test accuracy falls from its peak. PC follows a different trajectory. The controls in Section 3 show that PC can memorize more labels than it actually learns from the corrupted subset. Its restrained noise fitting therefore calls for an explanation in the learning process. Does inference produce targets that support learning from noisy labels, or does the advantage come from the inferred inputs used to learn those targets?

To distinguish these explanations, we use Forward Update, introduced for training deeper PC networks (Qi et al., 2026). This rule (PC+F) retains PC's inferred targets but replaces the inferred inputs with forward inputs during weight learning. Figure 1 contrasts BP's chain-rule update with PC and PC+F. If the inferred targets account for reduced noise memorization, this change should preserve it. Instead, PC+F memorizes more incorrect labels, identifying the input read by the weight update as a key part of PC's behavior. Intermediate-input rules and an error-inference sweep in ePC (Goemaere et al., 2025) further examine this role in Section 4.

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

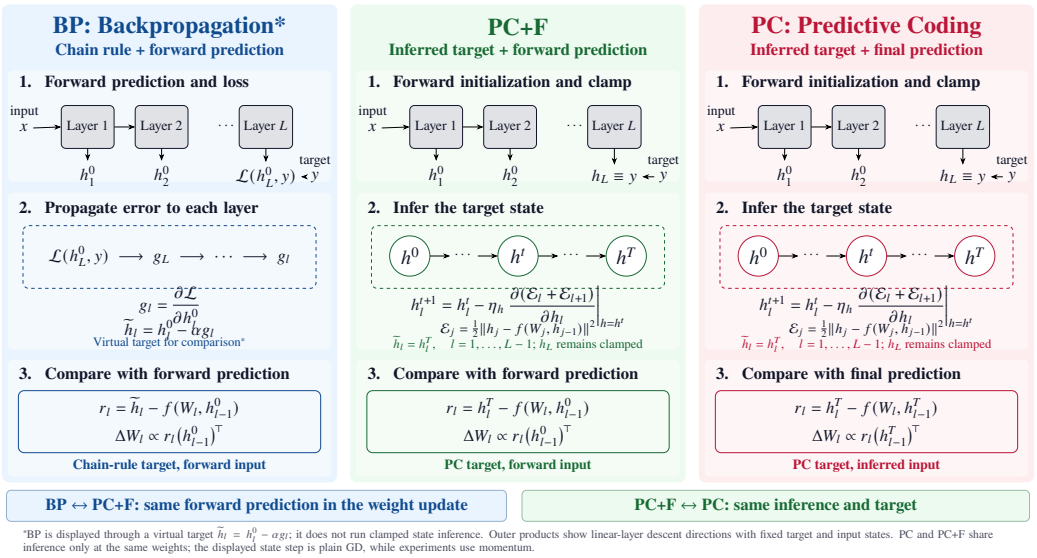


Figure 1: **State inference and weight-update inputs.** PC and PC+F use the same target-conditioned inference; their local updates read h^T and h^0, respectively. BP uses the chain rule.

How does the inferred input change the learning signal? It determines both the local prediction error and the parameter derivative. In Section 5, paired true/wrong probes at fixed images and weights show that hidden-state adaptation absorbs the wrong label's larger forward mismatch, leaving an output residual close to that of the true label. Reading the inferred input then produces a weaker and differently directed response to label replacement. Controlled target probes and a finite-step local model explain how this filtering depends on the direction of the target change.

We then ask how these filtered responses accumulate into memory. An optimizer comparison shows that the small raw gradients of PC do not imply small applied updates. In Appendix G.1, corruption-structure experiments examine the memory formed when incorrect associations recur across examples.

Related work. Benchmark studies establish the training behavior of PC (Pinchetti et al., 2025). Theoretical work describes PC learning as prospective configuration (Song et al., 2024), relates the inference equilibrium to target propagation and to a rescaled loss (Millidge et al., 2022c; Innocenti et al., 2024), and shows in linear networks that reading inferred presynaptic states makes single updates stable across learning rates (Alonso et al., 2022). We examine the same choice of state in trained nonlinear networks through target-specific memory. Appendix I gives the broader context.

Contributions.

- We establish that PC's reduced memorization of incorrect labels is a distinct learning behavior. PC fits clean examples and can learn fixed random labels, yet it retains fewer wrong labels during mixed training and does not follow BP's late decline. This contrast makes label noise a way to study how target information passes from state inference into shared weights. Section 3 develops this result.
- We separate the target produced by inference from the input read by the weight update. PC and PC+F share inference at a fixed checkpoint, but using the forward rather than the inferred input changes the label-induced gradient and increases noise memorization in training. A continuous mixed-input rule turns this interface into a controllable choice in the learning algorithm, as shown in Section 4.
- We derive a finite-step local operator for residual transfer and use controlled targets to expose its directional structure: inference strongly absorbs class contrasts while preserving a common output shift. Corruption experiments show that repeated, structured wrong-label responses can still be

learned. This connects selective transfer in one inference episode to the memory formed across training, from Section 5 to Appendix G.1.

2 PREDICTIVE CODING: STATE INFERENCE AND WEIGHT LEARNING

2.1 INFERENCE LEARNING AND WEIGHT UPDATE

Let $h_l^0 = f_l(h_{l-1}^0; W_l)$ be the feedforward states, with $h_0 = x$. Inference adjusts the free hidden states by descending

$$\mathcal{E}(h_{1:L-1}) \;=\; \sum_{l=1}^{L} \tfrac{1}{2}\big\|h_l - f_l(h_{l-1}; W_l)\big\|^2, \qquad h_0 = x, \quad h_L = y, \tag{1}$$

with the input and the output target clamped; after T steps the states are h^T. Two quantities describe what inference has changed at layer l:

$$\epsilon_l \;=\; h_l^T - f_l(h_{l-1}^T; W_l) \qquad \textit{residual}\text{: what the settled states still fail to predict,} \tag{2}$$

$$d_l \;=\; h_l^T - f_l(h_{l-1}^0; W_l) \;=\; h_l^T - h_l^0 \qquad \textit{displacement}\text{: how far inference moved the state.} \tag{3}$$

After inference, PC differentiates each local squared error with the stored states held fixed (Whittington & Bogacz, 2017; Song et al., 2020). Let $J_{W,l}^T$ be the Jacobian of the local prediction with respect to its parameters, evaluated at input h_{l-1}^T. The local gradient is

$$g_l^{\mathrm{PC}} = -(J_{W,l}^T)^\top \epsilon_l. \tag{4}$$

Thus inference affects weight learning through both the remaining residual and the input at which the parameter derivative is evaluated. The superscript T labels the final inference step; matrix transpose is written $\top$. We use g for a gradient, $-g$ for a descent direction, and u for the update applied by the optimizer. Appendix A.1 lists every symbol used in the paper. No derivative is taken through the inference trajectory when computing this local weight gradient. Section 4 makes the role of the input explicit by changing which stored state the rule reads.

2.2 LEARNING CONDITIONS AND BEHAVIORAL MEASUREMENTS

The central experiments use VGG5 networks on CIFAR data (Krizhevsky et al., 2009). We record clean fitting m_c on uncorrupted training images, noisy-label fitting m_n on corrupted images, recovery of the true label on those images c, and test accuracy. These measurements describe which relationships the trained network reproduces from a forward pass.

We evaluate both the training endpoint and the checkpoint selected by clean validation. The tables label the corresponding protocols R0 and ES: R0 averages final 10 epoches' test evaluations, while ES evaluates the validation-selected checkpoint. Detailed partitions, budgets, and statistical units are listed in Appendices A.2 and B.1.

3 FITTING ABILITY AND NOISE MEMORIZATION

Question. Does limited fitting ability explain PC's reduced noise memorization?

3.1 LEARNING TRAJECTORIES UNDER LABEL NOISE

We first train BP and PC on the same mixture of clean and corrupted labels and track what each network predicts as training continues. In Figure 2a–b, BP establishes clean fitting before increasingly fitting incorrect labels; its test accuracy rises and then falls, as in earlier memorization studies (Arpit et al., 2017; Toneva et al., 2019; Liu et al., 2020). PC continues to fit clean examples while its wrong-label fitting remains lower, true-label recovery stays higher, and test accuracy stays near its peak.

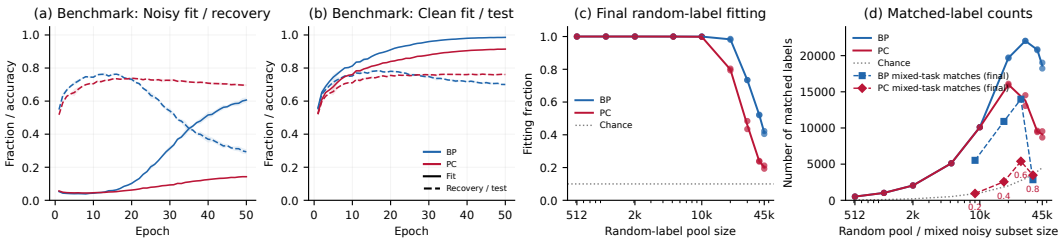


Figure 2: **Noisy-label training and random-label fitting.** (a,b) PC and BP training trajectories; solid lines show fit, dashed lines recovery or test accuracy. (c) Final random-label fitting across pool sizes; (d) matched-label counts and chance expectation. Random-label dots show seeds, with lines joining two-seed means. Mixed-task diamonds show three-seed final means; labels give noise ratios.

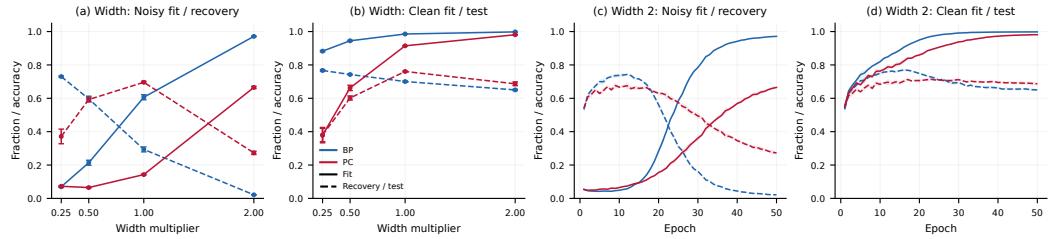


Figure 3: **Width endpoints and width-two trajectories.** (a,b) Final fit and recovery or test accuracy across width multipliers; (c,d) training trajectories at width two. Solid lines show fit, dashed lines recovery or test accuracy. Lines and bars summarize three seeds.

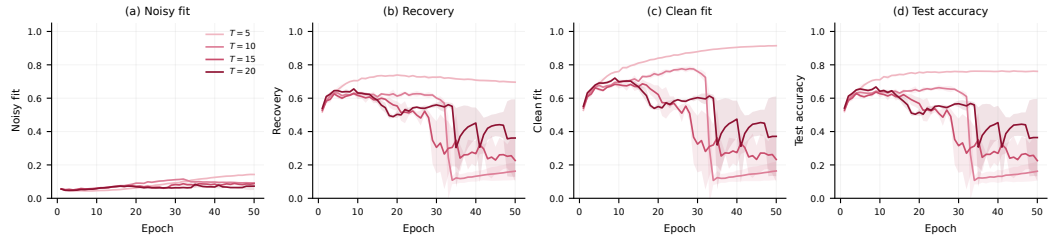


Figure 4: **Training trajectories across inference durations.** Width-one PC runs at four inference-step settings; lines and bands summarize three seeds.

We then run the experiments on more datasets: Fashion-MNIST (Xiao et al., 2017)/MLP5 repeats the main trajectory: BP test accuracy falls after its peak while PC retains clean fitting, low wrong-label fitting, and stable test accuracy. On CIFAR-100/VGG5, BP has yet to show a late decline and PC loses clean fitting and test accuracy under heavier corruption. Appendix B.4 shows both trajectories. On CIFAR-10N human annotations (Wei et al., 2022), PC again ends with far lower wrong-label fitting than BP in every annotation condition (Appendix G.3).

Now, from the results over CIFAR10 and Fashion-MNIST, the question is why PC learns fewer wrong associations from this mixture. Figure 2c–d sets up the first check by comparing mixed-task memory with fitting on a separate pool of entirely random labels (Zhang et al., 2017).

3.2 Random-label fitting ability

The random-label experiment asks whether PC simply lacks the ability to memorize arbitrary image–label pairs. We train PC and BP with MSE on fixed random labels over increasing image pools, using a common weight-update step budget. Figure 2c shows that both methods nearly fit the smaller pools, while BP fits more labels on larger pools. Thus capacity affects the comparison, but PC can still learn many arbitrary associations. Figure 2d makes the relevant comparison: PC's random-label task yields way more matched labels than the corrupted subset of its mixed-task runs. Appendix B.3.1 gives run lengths and protocols.

3.3 WIDTH AND INFERENCE DURATION

We next change two resources that could limit fitting in mixed training. Increasing width gives the network more parameters. Figure 3a–b shows that wider PC models do memorize more wrong labels and fit clean examples better. But the width-two trajectories in panels c–d show what happens over time: BP acquires wrong labels after clean fitting and falls from its test peak, whereas PC maintains a smaller wrong-label fit and a steadier test trajectory.

Increasing the inference duration gives PC more computation budget before each weight update. Figure 4 shows the opposite of a fitting gain: the longer-duration runs retain low wrong-label fitting while clean fitting, recovery, and test accuracy deteriorate late in training and vary across seeds. More inference computation makes these training configurations less stable; it does not produce stronger learning of either clean or corrupted targets. Appendix B.3 gives the configurations and random-label controls for these runs.

3.4 FILTERING MISCLASSIFIED CLEAN EXAMPLES

The mixed task may keep directing PC toward clean examples it has not yet learned. To test whether this competition holds back noise memory, we remove misclassified clean examples after clean fitting is established, while retaining correctly classified clean examples and every corrupted example. Appendix B.2 shows a modest increase in wrong-label fitting, alongside a loss of further clean learning. Noise memory remains low despite the continued exposure to all corrupted examples.

Answer. PC can memorize more arbitrary labels than it learns from the corrupted subset during mixed training. Width increases noise memory while preserving a trajectory distinct from BP's; longer inference destabilizes training, and filtering unfinished clean examples releases little additional noise memory. These results direct the explanation to PC's learning process: which part of it governs whether an incorrect target reaches shared weights?

4 CONTROLLING THE STATE-TO-WEIGHT INTERFACE

Question. Is the inferred target sufficient to preserve PC's reduced noise memorization, or does the input used to learn it matter?

4.1 SEPARATING INFERENCE FROM WEIGHT UPDATES

At layer l, PC and PC+F share the inferred target h_l^T. They differ in the input used to predict it: standard PC reads h_{l-1}^T. If the inferred target alone accounts for reduced noise fitting, retaining that target while changing the input should preserve the behavior. The forward-input rule PC+F instead recomputes the prediction at the saved state h_{l-1}^0, while retaining the inferred target state h_l^T. Because $f_l(h_{l-1}^0; W_l) = h_l^0$, its local error is the displacement d_l from Eq. (3) rather than the residual ϵ_l. Its gradient is therefore

$$g_l^{\mathrm{F}} = -(J_{W,l}^0)^\top d_l, \tag{5}$$

where $J_{W,l}^0$ is the parameter Jacobian at the forward input.

The output layer makes the two effects of this change easy to see. Write its prediction as $Vh_{L-1} + b$, with output state clamped to y. The last hidden state h_{L-1} supplies the input; it is free during inference. The weight gradients are

$$\begin{aligned} g_V^{\mathrm{PC}} &= -(y - Vh_{L-1}^T - b)(h_{L-1}^T)^\top, \\ g_V^{\mathrm{F}} &= -(y - Vh_{L-1}^0 - b)(h_{L-1}^0)^\top. \end{aligned} \tag{6}$$

Changing the input changes both the prediction error and the activity that multiplies it. The bias gradient is the negative residual.

Paired probes hold weights and inference results fixed. In separate training runs, Figure 5 shows the accumulated effect: PC+F and BP continue to acquire wrong labels after the validation-selected checkpoint, while PC retains lower endpoint wrong-label fitting.

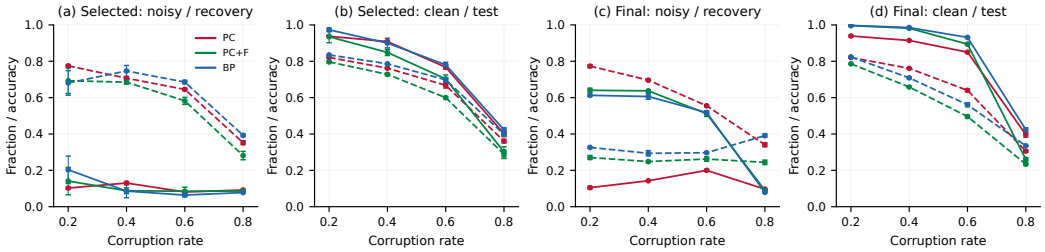


Figure 5: **Corruption-rate results at selected and final checkpoints.** (a,b) Clean-validation-selected checkpoints; (c,d) training endpoints. Solid lines show wrong-label or clean fitting; dashed lines show recovery or test accuracy. Points and bars summarize three seeds.

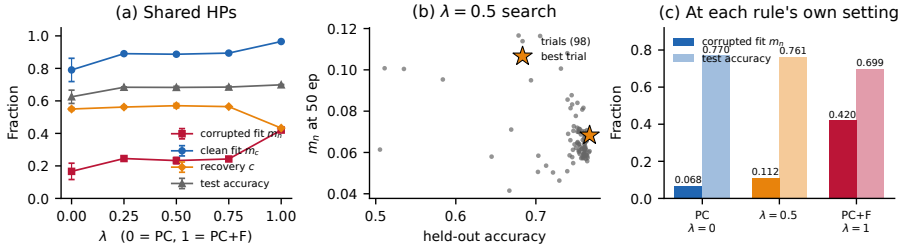


Figure 6: **Training outcomes for intermediate update rules.** (a) Shared-configuration λ sweep; (b) $\lambda = 0.5$ held-out-accuracy search, with the selected trial marked by a star; (c) separately tuned PC and midpoint runs alongside the shared-sweep PC+F endpoint. Appendix C.2 gives the training configurations.

4.2 Intermediate update rules

The PC/PC+F comparison changes the weight input at its two endpoints. We next ask how noise memory changes when the update reads a mixture of inferred and forward inputs. At each learned layer, including the readout, we keep the inferred target h_l^T and set the input to

$$h_{l-1}^{\lambda} = (1-\lambda)h_{l-1}^{T} + \lambda h_{l-1}^{0}, \qquad g_l^{\lambda} = -(J_{W,l}^{\lambda})^{\top}\left[h_l^T - f_l(h_{l-1}^{\lambda}; W_l)\right]. \tag{7}$$

The stored states are fixed during weight differentiation, and $h_0 = x$ is unchanged. Thus $\lambda = 0$ gives PC and $\lambda = 1$ gives PC+F. Each intermediate value recomputes both the local residual and its parameter Jacobian at h_{l-1}^{λ}; the rule does not interpolate two finished gradients.

Figure 6a varies λ under shared training settings. The successful interior runs retain high clean fitting and test accuracy while fitting substantially fewer wrong labels than the forward-input endpoint. Even a partial inferred input changes what the weights learn from corrupted targets. We then search the midpoint rule's training settings using held-out accuracy. Panel b shows the candidate runs and the selected trial, making visible the range of clean accuracy and wrong-label fitting obtainable with the same input mixture. Panel c compares the selected midpoint runs with a separately tuned PC run and the shared-sweep PC+F endpoint: the midpoint retains test accuracy close to PC while its wrong-label fitting stays far below PC+F. The input read by the update therefore provides a control over noise memory while retaining clean learning. Appendix C.2 gives the selection and training protocols; Appendix E.2 derives the rule.

4.3 Strengthening error inference in ePC

PC+F shows what happens when the weight rule stops reading inferred inputs. Error-based PC (ePC) (Goemaere et al., 2025) changes the strength of inference before those inputs are read. ePC uses backpropagation through the network to infer prediction-error variables e_l, then reconstructs hidden states as $h_l^T = f_l(h_{l-1}^T; W_l) + e_l^T$. Its weight rule differentiates each local prediction at the reconstructed input h_{l-1}^T. For a hidden layer in the tested regime $Te_{\mathrm{lr}} < 1$, the implementation's loss scaling gives

$$g_l^{\mathrm{ePC}} = -(J_{W,l}^{T})^{\top}\frac{e_l^T}{Te_{\mathrm{lr}}}. \tag{8}$$

At small e_{lr}, the normalized error approaches the negative backward sensitivity at the forward state, while h_{l-1}^{T} approaches h_{l-1}^{0}. Larger inference steps move the update away from this forward-state regime: inferred errors change the local learning signal and the input at which it is applied.

We vary e_{lr} at fixed inference iteration count and fixed nominal weight-optimizer settings within each series. Wrong-label fitting decreases as e_{lr} rises through the range in which every run completes. Figure 7 shows that clean fitting remains high. At the next step size, completed runs show a much larger drop in wrong-label fitting, together with lower clean fitting and reduced training stability. The ePC trajectory extends the PC+F result: noise memory changes when inference has a stronger role in forming the local update. Full trajectories, success counts, and the loss normalization appear in Appendix H.2.1.

Answer. At matched inference, replacing the inferred input with the forward input increases noise memory. Intermediate inputs change it continuously; changing error inference in ePC also changes the subsequent local learning trajectory. The state presented to the weight rule is a central part of PC's reduced noise memorization. What does reading that state change in the learning signal?

5 Filtering target changes through inference

Question. After inference, what remains of a target change, and how does it reach the weights?

5.1 Where the wrong-target mismatch goes

At frozen checkpoints, we measure the initial mismatch relative to the inferred residual and the corresponding complete readout-gradient ratio:

$$A(y) = \frac{\|d_L(y)\|}{\|\epsilon_L(y)\|}, \qquad G(y) = \frac{\|g_L^{\mathrm{PC}}(y)\|}{\|g_L^{\mathrm{F}}(y)\|}. \tag{9}$$

Both rules are evaluated after shared inference, including readout weights and bias. At PC-trained checkpoints, the wrong label creates a larger forward mismatch, yet true and wrong targets leave residuals of similar magnitude. Figure 8a–b shows the trajectories, and Table 1 gives the paired measurements. Inference absorbs both mismatches. The complete update also depends on the inferred input, examined below. The absorption contrast grows over training while G falls. Table 8 also shows that PC+F-trained checkpoints retain similar forward and residual scales.

Proposition 1 (Prediction change during inference)**.** *At fixed weights and inferred states,*

$$d_l = \epsilon_l + q_l, \qquad q_l = f_l(h_{l-1}^{T}; W_l) - f_l(h_{l-1}^{0}; W_l). \tag{10}$$

If f_l is differentiable at h_{l-1}^{0}, then $q_l = J_{h,l} d_{l-1} + o(\|d_{l-1}\|)$, where $d_{l-1} = h_{l-1}^{T} - h_{l-1}^{0}$ is the displacement of the input and $J_{h,l}$ is the input Jacobian at h_{l-1}^{0}.

At the clamped output, q_L is the prediction change caused by hidden-state motion. The small residual means this change accounts for nearly all of the initial wrong-target mismatch. True-target mismatch is absorbed too. Native clean/noisy pools retain this contrast; re-inferring noisy images with their true labels moves it toward the clean-pool values, as shown in Figure 8c. Appendix D.3 gives the paired control.

5.2 From absorbed mismatch to a weight response

We fix the image and weights, infer under true and wrong labels, and compare the label-induced change under rules $a \in \{\mathrm{PC}, \mathrm{F}\}$:

$$\Delta g_i^{a} = g_i^{a,n} - g_i^{a,c}. \tag{11}$$

$$\rho_i = \frac{\|\Delta g_i^{\mathrm{PC}}\|}{\|\Delta g_i^{\mathrm{F}}\|}, \qquad C_i^{\Delta} = \cos(\Delta g_i^{\mathrm{PC}}, \Delta g_i^{\mathrm{F}}). \tag{12}$$

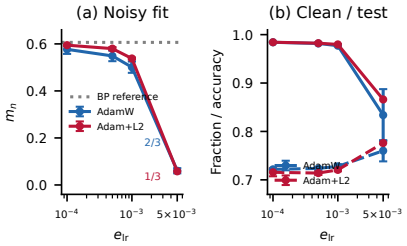


Figure 7: **ePC step-size sweep.** (a) Noisy fit; (b) clean fit (solid) and test accuracy (dashed). Full grid in Table 24.

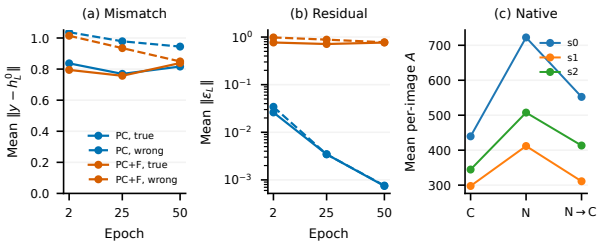


Figure 8: **Target mismatch and absorption.** (a,b) Mean forward mismatch $\|d_L\| = \|y - h_L^0\|$ and residual $\|\epsilon_L\|$: true targets solid, wrong dashed. (c) Native-pool contrast A; N→C re-infers noisy images with true labels.

Target	Mean $\|d_L\|$	Mean $\|\epsilon_L\|$	Mean A	Mean G
True	0.76915	0.003454	474.76	0.004834
Wrong	0.97871	0.003454	546.06	0.003675

Table 1: Paired true and wrong targets at a frozen PC-trained checkpoint. Values average per-image measurements from the same corrupted-image pool at epoch 25. A and G are formed per image before averaging; the complete checkpoint series appears in Table 8.

At PC-trained checkpoints, the readout response is strongly attenuated and its mean cosine becomes negative. Figure 9a–c shows that earlier layers differ less. PC+F-trained checkpoints retain close alignment. These are raw gradient comparisons; Appendix F.1 examines the steps applied by the optimizer.

Let $h_c = h_{L-1}^T(y^c)$, $\Delta h = h_{L-1}^T(y^n) - h_c$, and $\Delta\epsilon = \epsilon_L(y^n) - \epsilon_L(y^c)$. With $\epsilon_c = \epsilon_L(y^c)$, the exact readout change is

$$\Delta g_V^{\mathrm{PC}} = \underbrace{-\Delta\epsilon\, h_c^\top}_{t_1} + \underbrace{-\epsilon_c\, \Delta h^\top}_{t_2} + \underbrace{-\Delta\epsilon\, \Delta h^\top}_{t_3}, \qquad \Delta g_b^{\mathrm{PC}} = -\Delta\epsilon. \tag{13}$$

The terms change the residual, input, and both jointly. The residual and input terms are nearly orthogonal, while the joint term often opposes the input term, as shown in Figure 9d–e. Their combination changes the response's magnitude and direction. General Jacobian identities and the differential form appear in Appendix E.1.10.

5.3 Direction-dependent residual transfer

A fixed local model $r(z, w) = r_0 - M_z z - M_w w$ separates hidden-state displacements z from parameter changes w; here r stacks the residuals of all layers, r_0 is its value at the forward states, and M_z, M_w are its Jacobians with respect to states and parameters. During inference, $w = 0$.

Proposition 2 (Finite-step residual transfer)**.** *Keep M_z, M_w fixed, initialize $z_0 = 0$, and take T ordinary gradient-descent steps of size η_h on $\mathcal{E}(z) = \frac{1}{2}\|r_0 - M_z z\|^2$. With $H = M_z^\top M_z$,*

$$P_T = M_z H^\dagger [I - (I - \eta_h H)^T] M_z^\top, \tag{14}$$

$$r_T = (I - P_T) r_0, \qquad g_w = -M_w^\top (I - P_T) r_0. \tag{15}$$

Here $H^\dagger$ is the Moore–Penrose pseudoinverse. If every positive eigenvalue μ_j of H satisfies $0 < \eta_h \mu_j < 2$, then P_T converges to the orthogonal projector $M_z H^\dagger M_z^\top$ onto $\mathrm{col}(M_z)$.

The operator $I - P_T$ retains different fractions along different residual directions. Equation (121) shows that scaling one direction scales its response proportionally. For the selector O that extracts the output block ($OO^\top = I$) and a unit output direction v, the predicted residual fraction is

$$\pi_{L,T}(v) = \|O(I - P_T)O^\top v\|. \tag{16}$$

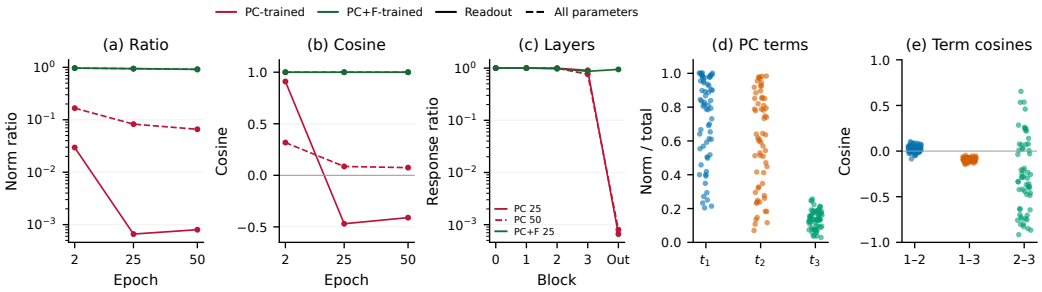


Figure 9: **Label-replacement gradient responses and readout terms.** (a,b) PC/PC+F norm ratios and cosines of Δg across checkpoints; solid lines denote readout and dashed lines all parameters. (c) Ratios by layer. (d,e) Relative norms and pairwise cosines of the readout terms t_1 (residual), t_2 (input), and t_3 (joint change) from Eq. (13).

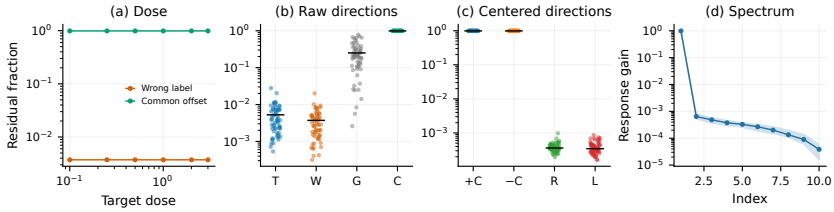


Figure 10: **Target-dose and direction probes.** (a) Residual fraction across doses; (b,c) unit-dose fractions for raw and centered directions; (d) singular values of the coordinate-response map. T/W/G/C denote true-label, wrong-label, Gaussian, and common-offset drives; $+C/-C$ are opposite common-offset drives, and R/L are centered random and label drives. Gaussian and common-offset drives match the wrong-label displacement norm.

We test this prediction at fixed images and weights with targets $y_a = h_L^0 + av$. Varying a at fixed direction leaves the residual fraction nearly unchanged. Figure 10a–b shows that changing direction at matched norm changes it.

A common output shift along $\bar{\mathbf{1}} = \mathbf{1}/\sqrt{K}$ survives inference, whereas centered drives Qv, $Q = I - \bar{\mathbf{1}}\bar{\mathbf{1}}^\top$, are strongly attenuated. This explains the larger response to full-space Gaussian drives. Label replacements lie in the centered subspace, as shown in Figure 10c. A response matrix reconstructed from coordinate probes has one high-gain mode and weak remaining modes (panel d). Vector checks and hidden-state clamps further characterize this response in Appendices D.2 and D.3.

Answer. Paired states, gradients, and target probes show how hidden-state adaptation absorbs mismatch and filters the label-induced weight response. The filter depends on target direction and on the input read by the local rule. Appendix F.1 follows the resulting gradients through applied updates, and Appendix G.1 examines how repeated wrong-label responses combine across examples.

6 Conclusion

Inference shapes a PC weight update by changing both the target mismatch left for local learning and the state at which that mismatch becomes a parameter gradient. The paired true/wrong probes show this transformation: an incorrect label creates a larger forward mismatch, yet inference absorbs the extra displacement and leaves a readout residual close to that of the true label. PC+F turns the same inference result into a stronger label-induced update and greater noise memory; intermediate rules and the ePC sweep show how training changes as inference contributes more to local learning. Repeated, structured wrong-label responses can still build shared memory. Together, these results make the state-to-weight interface a concrete design variable and motivate testing target-conditioned state adjustment before parameter updates in other learning rules.

AI USE STATEMENT

In this work, we used generative AI tools to polish the writing, to make light edits to code, and to assist in proving mathematical claims in Appendix E. The code edits covered visualization and repeated multi-seed runs based on our existing experiment code.

REPRODUCIBILITY STATEMENT

Experimental protocols and aggregation conventions are specified in Appendix A.2; numerical validation and its scope are described in Appendix E.3. The analysis distinguishes sample variation from variation across training seeds. The code will be made publicly available upon acceptance."

REFERENCES

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

Nick Alonso, Beren Millidge, Jeffrey Krichmar, and Emre O Neftci. A theoretical framework for inference learning. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://papers.nips.cc/paper_files/paper/2022/hash/f242c4cba2467637256722cb679642bd-Abstract-Conference.html.

Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. A closer look at memorization in deep networks, 2017.

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

Rafal Bogacz. A tutorial on the free-energy framework for modelling perception and learning. *Journal of Mathematical Psychology*, 76:198–211, 2017.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: evaluating and testing unintended memorization in neural networks. In *USENIX Security Symposium*, 2019.

Vitaly Feldman. Does learning require memorization? A short tale about a long tail. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. URL https://arxiv.org/abs/1906.05271.

Karl Friston. The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience*, 11(2): 127–138, 2010.

Cédric Goemaere, Gaspard Oliviers, Rafal Bogacz, and Thomas Demeester. Error optimization: Overcoming exponential signal decay in deep predictive coding networks. *arXiv preprint arXiv:2505.20137*, 2025.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems*, 31, 2018.

Francesco Innocenti, Ryan Singh, and Christopher L. Buckley. Understanding predictive coding as an adaptive trust-region method. *arXiv preprint arXiv:2305.18188*, 2023. URL https://arxiv.org/abs/2305.18188.

Francesco Innocenti, El Mehdi Achour, Ryan Singh, and Christopher L. Buckley. Only strict saddles in the energy landscape of predictive coding networks? In *Advances in Neural Information Processing Systems*, volume 37, pp. 53649–53683, 2024. URL https://arxiv.org/abs/2408.11979.

Francesco Innocenti, El Mehdi Achour, and Rafal Bogacz. On the infinite width and depth limits of predictive coding networks. In *International Conference on Machine Learning*, 2026a.

Francesco Innocenti, El Mehdi Achour, and Christopher L Buckley. μ pc: Scaling predictive coding to 100+ layer networks. *Advances in Neural Information Processing Systems*, 38:87414–87455, 2026b.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

Dong-Hyun Lee, Saizheng Zhang, Asja Fischer, and Yoshua Bengio. Difference target propagation. In *Proc. ECMLPKDD*, 2015.

Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. *arXiv preprint arXiv:2002.07394*, 2020a.

Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In *International Conference on Artificial Intelligence and Statistics*, pp. 4313–4324, 2020b.

Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Pratyush Maini, Michael C. Mozer, Hanie Sedghi, Zachary C. Lipton, J. Zico Kolter, and Chiyuan Zhang. Can neural network memorization be localized? In *International Conference on Machine Learning*, 2023.

Alexander Meulemans, Francesco S. Carzaniga, Johan A. K. Suykens, João Sacramento, and Benjamin F. Grewe. A theoretical framework for target propagation, 2020.

Alexander Meulemans, Nicolas Zucchet, Seijin Kobayashi, Johannes von Oswald, and João Sacramento. The least-control principle for local learning at equilibrium. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Beren Millidge, Alexander Tschantz, and Christopher L Buckley. Predictive coding approximates backprop along arbitrary computation graphs. *arXiv:2006.04182*, 2020.

Beren Millidge, Tommaso Salvatori, Yuhang Song, Rafal Bogacz, and Thomas Lukasiewicz. Predictive coding: Towards a future of deep learning beyond backpropagation? *arXiv preprint arXiv:2202.09467*, 2022a.

Beren Millidge, Yuhang Song, Tommaso Salvatori, Thomas Lukasiewicz, and Rafal Bogacz. Backpropagation at the infinitesimal inference limit of energy-based models: unifying predictive coding, equilibrium propagation, and contrastive hebbian learning. *arXiv preprint arXiv:2206.02629*, 2022b.

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*, 2022c.

Nagarajan Natarajan, Inderjit S. Dhillon, Pradeep Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2013.

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

Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1944–1952, 2017.

Luca Pinchetti, Chang Qi, Oleh Lokshyn, Cornelius Emde, Amine M'Charrak, Mufeng Tang, Simon Frieder, Bayar Menzat, Gaspard Oliviers, Rafal Bogacz, et al. Benchmarking predictive coding networks–made simple. *International Conference on Learning Representations*, 2025:35701–35734, 2025.

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

Chang Qi, Matteo Forasassi, Thomas Lukasiewicz, and Tommaso Salvatori. Towards the training of deeper predictive coding neural networks. In *Forty-third International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2506.23800.

Rajesh P. N. Rao and Dana H. Ballard. Predictive coding in the visual cortex: A functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2(1):79–87, 1999.

Robert Rosenbaum. On the relationship between predictive coding and backpropagation. *PLOS ONE*, 17(3):e0266102, 2022.

Tommaso Salvatori, Yuhang Song, Yujian Hong, Lei Sha, Simon Frieder, Zhenghua Xu, Rafal Bogacz, and Thomas Lukasiewicz. Associative memories via predictive coding. In *Advances in Neural Information Processing Systems*, volume 34, 2021a.

Tommaso Salvatori, Yuhang Song, Thomas Lukasiewicz, Zhenghua Xu, and Rafal Bogacz. Predictive coding can do exact backpropagation on convolutional and recurrent neural networks. *arXiv:2103.03725*, 2021b.

Tommaso Salvatori, Yuhang Song, Beren Millidge, Zhenghua Xu, Lei Sha, Cornelius Emde, Rafal Bogacz, and Thomas Lukasiewicz. Incremental predictive coding: A parallel and fully automatic learning algorithm. *International Conference on Learning Representations 2024*, 2024.

Tommaso Salvatori, Ankur Mali, Christopher L Buckley, Thomas Lukasiewicz, Rajesh PN Rao, Karl Friston, and Alexander Ororbia. A survey on neuro-mimetic deep learning via predictive coding. *Neural networks*, pp. 108161, 2025.

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

Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. Learning from noisy labels with deep neural networks: A survey. *IEEE transactions on neural networks and learning systems*, 34(11):8135–8153, 2022.

Yuhang Song, Thomas Lukasiewicz, Zhenghua Xu, and Rafal Bogacz. Can the brain do backpropagation? — Exact implementation of backpropagation in predictive coding networks. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

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*, pp. 1–11, 2024.

Cory Stephenson, Suchismita Padhy, Abhinav Ganesh, Yue Hui, Hanlin Tang, and SueYeon Chung. On the geometry of generalization and memorization in deep neural networks. In *International Conference on Learning Representations*, 2021.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1812.05159.

Jiaheng Wei, Zhaowei Zhu, Hao Cheng, Tongliang Liu, Gang Niu, and Yang Liu. Learning with noisy labels revisited: A study using real-world human annotations. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2110.12088.

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), 2017.

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

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Sy8gdB9xx.

Nicolas Zucchet and João Sacramento. Beyond backpropagation: bilevel optimization through implicit differentiation and equilibrium propagation. *Neural Computation*, 34(12):2309–2346, 2022.

APPENDIX CONTENTS

A Notation, algorithms, and measurement conventions . . . 14
A.2 Experimental protocols . . . 14
B Noise trajectories, width, and inference budget . . . 16
B.1 Training archive and selection protocols . . . 16
B.2 Filtering misclassified clean examples . . . 16
B.3 Width, inference budget, and random-label fitting . . . 18
B.4 Training trajectories on Fashion-MNIST and CIFAR-100 . . . 19
C PC/PC+F, intermediate rules, and paired updates . . . 20
C.1 Target-conditioned absorption across layers . . . 20
C.2 Intermediate-rule training . . . 22
D.4 Readout residual under longer inference at fixed weights . . . 26
G.1.1 Aggregation and coherence . . . 35
D Target adaptation, direction probes, and hidden clamps . . . 22
D.1 Controlled target scaling and direction . . . 22
D.2 Centered response maps and re-inferred label changes . . . 23
D.3 Native targets and hidden-input interventions . . . 24
E Theory and step-by-step derivations . . . 27
E.1 Detailed derivations of state-to-weight transfer . . . 27
E.2 Interpolation geometry and local control . . . 33
E.3 Numerical checks . . . 34
F Applied updates and layer-scaled BP . . . 34
F.1 Small local errors and applied updates . . . 34
G.2 Continuation and corruption-structure follow-ups . . . 36
F.2 Layer-scaled BP . . . 35
G Noise types and corruption rates . . . 35
G.1 Noise structure and cross-example aggregation . . . 35
G.3 Human-annotation results . . . 37
H PC variants and optimization settings . . . 38
H.1 Benchmark trajectories of PC variants . . . 38
H.2 ePC, PC+SF, and optimization settings . . . 38
H.2.1 Error-inference step-size sweep . . . 39
I Related work . . . 40
J Discussion . . . 41
K Limitations . . . 42

A NOTATION, ALGORITHMS, AND MEASUREMENT CONVENTIONS

A.1 NOTATION

Each symbol has one meaning throughout the paper. Table 2 collects them; derivation-local quantities, such as coefficients in a single proof, are defined where they appear.

A.2 EXPERIMENTAL PROTOCOLS

A.2.1 LOCAL LEARNING PROCEDURE

For each example, we first save a forward pass. We then clamp the output to the supplied target and run the prescribed state solver with fixed weights. The local weight derivative treats all stored states as constants. PC predicts each target state from its inferred input; PC+F predicts it from the saved forward input. A paired probe evaluates both derivatives before any weight update. A training run applies its selected rule and repeats the procedure for the next batch.

Symbol	Meaning
Network, data and inference	
x, y; y^c, y^n	input and clamped output target; true and wrong (corrupted) label
L, l; K	number of learned layers, layer index; number of classes
$f_l(\cdot; W_l)$; V, b	prediction of layer l; readout weight and bias ($f_L(h) = Vh + b$)
h_l^0; h_l^T	forward (feedforward) state; state after T inference steps
$\mathcal{E}$; η_h, m	prediction energy (Eq. (1)) or its local model; inference step size, momentum
e_l, e_{lr}	ePC error variables and error-inference step size (used only for ePC)
Local signals and gradients	
ϵ_l	*residual* after inference, $h_l^T - f_l(h_{l-1}^T; W_l)$ (Eq. (2)); "residual" always means ϵ
d_l	*displacement*, $h_l^T - h_l^0$ (Eq. (3)); at the output, $d_L = y - h_L^0$ is the forward mismatch
q_l	prediction change caused by the input displacement, $d_l = \epsilon_l + q_l$ (Prop. 1)
$J_{W,l}^T$, $J_{W,l}^0$; $J_{h,l}$	parameter Jacobian at the inferred / forward input; input Jacobian
g_l^{PC}, g_l^{F}, g_l^{λ}; λ	local gradients read at h_{l-1}^T, h_{l-1}^0, or the mixture h_{l-1}^{λ}; mixing coefficient
u	update applied by the optimizer (after AdamW or SGD scaling)
$\Delta(\cdot)$	change caused by replacing y^c with y^n at fixed image and weights (Δg, Δh, $\Delta\epsilon$, Δy)
t_1, t_2, t_3	residual, input and joint terms of Δg_V^{PC} (Eq. (13))
Measured ratios	
$A(y)$; $G(y)$	absorption contrast $\|d_L\|/\|\epsilon_L\|$; readout gradient ratio $\|g_L^{\mathrm{PC}}\|/\|g_L^{\mathrm{F}}\|$ (Eq. (9))
ρ; C^{Δ}	norm ratio and cosine of Δg^{PC} and Δg^{F} (Eq. (12))
$\pi_{L,T}(v)$; a, v	output residual fraction for a unit output drive v; dose, with target $y = h_L^0 + av$
$\bar{\mathbf{1}}$; Q	common output direction $\mathbf{1}/\sqrt{K}$; projector $I - \bar{\mathbf{1}}\bar{\mathbf{1}}^\top$ onto centered (class-contrast) directions
κ	coherence of per-example gradients g_i (Appendix G.1.1)
Local model (Proposition 2)	
r, r_0; z, w	stacked residual of all layers, its value at the forward states; state and parameter changes
M_z, M_w; H	Jacobians of r with respect to z and w; $H = M_z^\top M_z$
P_T; μ_j, ϕ_j	finite-step absorption operator; eigenvalues of H and left singular vectors of M_z
O	selector that extracts the output block of r ($OO^\top = I$)
Γ	output rescaling $I + \sum_\ell W_{L:\ell} W_{L:\ell}^\top$ of equilibrated linear PC networks (Appendix I)
Behavioral metrics	
m_c, m_n, c	fit of clean labels, fit of wrong labels on corrupted images, recovery of the true label on corrupted images
R0, ES	mean of the final ten test evaluations; clean-validation-selected checkpoint

Table 2: Notation.

A.2.2 PAIRED STATE AND GRADIENT PROBES

The mechanism experiment evaluates true and wrong targets on the same fixed pool of corrupted images. A fresh inference optimizer starts from the stored forward state for each target. After inference, the two weight rules use the same snapshot. Weight fingerprints and forward-state equality are checked during probing. Inputs are unaugmented for this evaluation.

Item	Recorded protocol
Network / activation	VGG5 / GELU; no batch normalization or dropout
Paired sample pool	200 corrupted images, one training seed per family
Checkpoint epochs	PC and PC+F: 2, 25, 50; intermediate: 25
True-class counts	24, 16, 16, 17, 20, 24, 24, 15, 19, 25
Batch / individual state check	Maximum relative discrepancy 8.1×10^{-5}
Shared endpoint replication	3 training seeds
Training augmentation	Random translations and orientation flips

Table 3: Protocol of the shared-configuration mechanism study. Sample variation in a paired probe and variation across independently trained models are reported separately.

Setting	Weight lr / decay	State lr / momentum	Steps / batch
Shared endpoint training	4.895×10^{-4} / 5.307×10^{-4}	1.0082×10^{-2} / 0.6	7 / 128

Table 4: Shared configuration used by the endpoint experiments. The weight schedule uses a 25-epoch cosine schedule followed by its terminal learning rate; the output is hard clamped. The fork's SGD learning rate is separately calibrated.

A.2.3 WHOLE-MODEL RECONSTRUCTION AND AGGREGATION

For disjoint parameter blocks l, the whole-model quantities follow from the blockwise norms and the blockwise cosines C_l^{Δ}:

$$\rho_{\mathrm{all}} = \frac{\left(\sum_l \|\Delta g_l^{\mathrm{PC}}\|^2\right)^{1/2}}{\left(\sum_l \|\Delta g_l^{\mathrm{F}}\|^2\right)^{1/2}}, \qquad C_{\mathrm{all}}^{\Delta} = \frac{\sum_l \|\Delta g_l^{\mathrm{PC}}\| \, \|\Delta g_l^{\mathrm{F}}\| \, C_l^{\Delta}}{\|\Delta g^{\mathrm{PC}}\| \, \|\Delta g^{\mathrm{F}}\|}. \tag{17}$$

We reconstruct these quantities per image and then summarize across images. Means of ratios, ratios of mean norms, and ratios of norms of means are distinct summaries. The transfer figure uses the first; aggregation probes record the last separately. The original layerwise records and the reconstruction identities make the plotted whole-model quantities reproducible.

B NOISE TRAJECTORIES, WIDTH, AND INFERENCE BUDGET

These experiments support Section 3: the benchmark configuration, sample filtering, and the capacity-related controls.

B.1 TRAINING ARCHIVE AND SELECTION PROTOCOLS

The behavioral tables select a single activation-consistent benchmark campaign. Each method retains its recorded configuration, with independent training seeds aggregated within a cell. Repeated exports are deduplicated by run identity. The mechanism report includes training runs already present in the experiment report, together with separate analysis artifacts.

B.2 FILTERING MISCLASSIFIED CLEAN EXAMPLES

This experiment asks whether PC starts fitting incorrect labels more strongly when clean examples that still require classification learning are removed from subsequent batches.

Method	Weight lr	Inference lr	Momentum	T	Weight input / timing
BP	$2.7882\,10^{-4}$	–	–	0	forward / post
PC	$1.0000\,e^{-4}$	0.0100	0.99	5	inferred / post
PC+F	$4.8950\,e^{-4}$	0.0101	0.60	7	forward / post
PC+SF	$4.6233\,e^{-4}$	0.2339	0.30	7	forward / post
ePC (Adam+L2)	$4.7084\,e^{-4}$	0.0010	–	5	error-based / post

Table 5: Benchmark and matched-ePC configurations, GELU VGG5/CIFAR-10 at symmetric corruption 0.4. Training lasts 50 epochs. Batch size is 128 for BP, PC, PC+F, and PC+SF, and 256 for ePC. The ePC row uses Adam with coupled L_2 weight decay; its inference rate is the error-state learning rate. Other rows use AdamW. The shared-weight mechanism campaign uses Table 4 instead.

Central benchmark convention	Definition
Training / held-out training split	45k / 5k
Standard test-set partition	30% clean validation / 70% reported evaluation
Endpoint training metrics	Exported endpoint for each run
Across-seed spread	Sample SD, denominator $n - 1$

Table 6: Metric conventions for the central benchmark. The shared-configuration mechanism study retains its own protocol.

B.2.1 Protocol

The width-one PC reference uses GELU VGG5/CIFAR-10 with symmetric corruption 0.4, batch size 128, and a 50-epoch warmup-cosine schedule. The weight learning rate is 10^{-4}, weight decay is 10^{-5}, state learning rate is 0.01, and state inference uses Nesterov momentum 0.99 for $T = 5$. Once clean fitting reaches $m_c \geq 0.8$, the retained set is rebuilt every epoch from currently correctly classified clean examples and all corrupted examples. Both filtered and reference runs use three seeds. Evaluation continues to use the complete original training set.

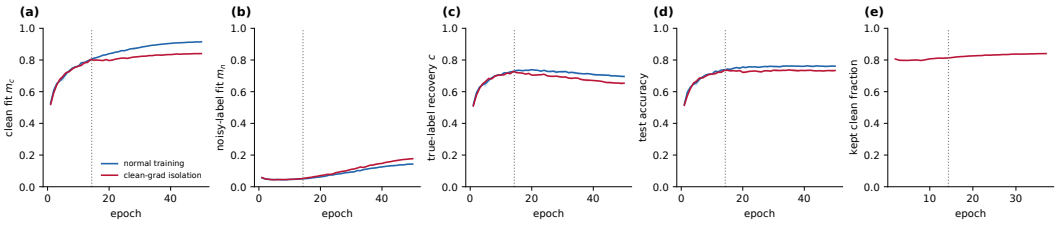


Figure 11: **Training trajectories after filtering misclassified clean examples.** (a–d) Clean fitting, wrong-label fitting, recovery, and test accuracy on the original training set. The vertical dashed line marks the filtering trigger; lines and bands summarize three seeds.

B.2.2 Results and interpretation

The filtered run retains low wrong-label fitting, with a modest increase relative to ordinary training. Clean fitting stays near the trigger level, and test accuracy decreases as misclassified clean examples lose further training exposure. Removing these examples changes the opportunity to learn clean targets without producing a large release of noise memorization. Correctly classified retained examples can still contribute gradients under the squared-error objective. The intervention therefore measures the effect of sample selection; the fixed-state comparison in Section 5.2 isolates the weight rule itself.

B.3 WIDTH, INFERENCE BUDGET, AND RANDOM-LABEL FITTING

B.3.1 RANDOM-LABEL POOL-SIZE SWEEP

The new sweep trains VGG5 with GELU on fixed random labels over increasing image pools. BP uses MSE and its selected weight hyperparameters; PC uses its reference noisy-task configuration. The batch size is fixed and the planned number of epochs scales inversely with pool size, targeting the weight-update step count of the main training run. This matches an outer update budget, while PC also performs state inference within each step. Small-pool runs end early; Table 7 records both planned and observed epoch counts.

Method	Pool size	Final fit	Matched labels	Planned epochs	Recorded epochs
BP	512	1.0000	512	4388	62/62
BP	1024	1.0000	1024	2194	62/62
BP	2048	1.0000	2048	1097	62/62
BP	5120	0.9998	5119	439	64/64
BP	10112	0.9997	10109	223	75/75
BP	20032	0.9828	19688	113	113/113
BP	30016	0.7342	22038	75	75/75
BP	39936	0.5212	20814	57	57/57
BP	44992	0.4140	18625	50	50/50
PC	512	1.0000	512	4388	62/62
PC	1024	1.0000	1024	2194	62/62
PC	2048	1.0000	2048	1097	62/62
PC	5120	0.9997	5118	439	64/64
PC	10112	0.9985	10097	223	75/75
PC	20032	0.7986	15998	113	113/113
PC	30016	0.4594	13789	75	75/75
PC	39936	0.2385	9524	57	57/57
PC	44992	0.2026	9115	50	50/50

Table 7: Random-label fitting under a common weight-update step budget. Fit and matched-label count are means over two seeds. Recorded epoch counts list each seed separately; small pools terminate early. Chance expectation is one tenth of pool size.

PC nearly fits the smaller pools and fits fewer labels than BP on larger pools. Figure 2 uses the final recorded fitting, including runs that still improve near the end. The mixed-task reference plots each symmetric noise ratio separately: the horizontal coordinate is the three-seed mean noisy-subset size, and the vertical coordinate is the mean of each seed's noisy-subset size times its final noisy fit. It compares demonstrated fitting across training tasks; clean examples and image repetition rates differ between those tasks.

B.3.2 FITTING UNDER DIFFERENT LEARNING CONFIGURATIONS

Figure 12 shows recorded random-label fitting counts with the chance expectation. Its width-two PC point uses the axhp2 configuration that reaches the reported high wrong-label fit in the matching mixed-label run. The width-one PC configuration with the shorter inference budget fits well above chance. The longer-budget configurations remain near chance and also lose clean fitting in the mixed-label runs. These outcomes track the stability of training as inference work increases. Width also changes the BP outcomes, with substantial variation across training seeds.

The recorded quantity is $M = Na_{\mathrm{random}}$, the number of correct predictions in the random-label evaluation pool. With uniformly random labels over K classes, the chance expectation is N/K. This baseline is necessary for interpreting the count: values near it describe near-chance prediction under the training protocol. Above-chance fitting shows that the configured learner acquires some of the fixed random associations.

The mixed-label trajectories show what the additional inference work buys. The shorter-budget configuration keeps its clean fit while wrong-label fitting remains restrained. The longer-budget configurations lose clean fit late in training without acquiring more wrong labels. The random-label

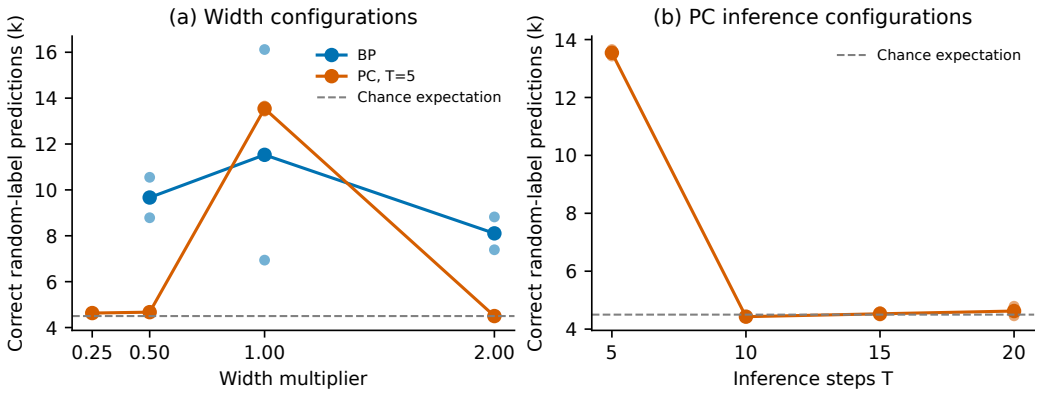


Figure 12: **Random-label fitting across width and inference budget.** (a) Width configurations; (b) width-one PC at different T. Dots are individual seeds, lines join two-seed means, and the dashed line marks chance expectation. The width-two PC point uses the axhp2 configuration matched to the mixed-label width-two run.

results follow the same training pattern: the stable configuration learns fixed associations, whereas the deteriorating configurations finish near chance.

B.4 Training trajectories on Fashion-MNIST and CIFAR-100

On Fashion-MNIST/MLP5, the moderate-corruption runs repeat the CIFAR-10 trajectory: BP first learns the clean examples, then its test accuracy falls from its peak as wrong-label fitting grows. PC retains clean fitting while wrong-label fitting stays low, and its test trajectory remains near its peak, as shown in Figure 13. At the highest corruption rate, both methods lose clean performance and terminate early. These runs use squared-error PC and cross-entropy BP, with GELU activations in both models.

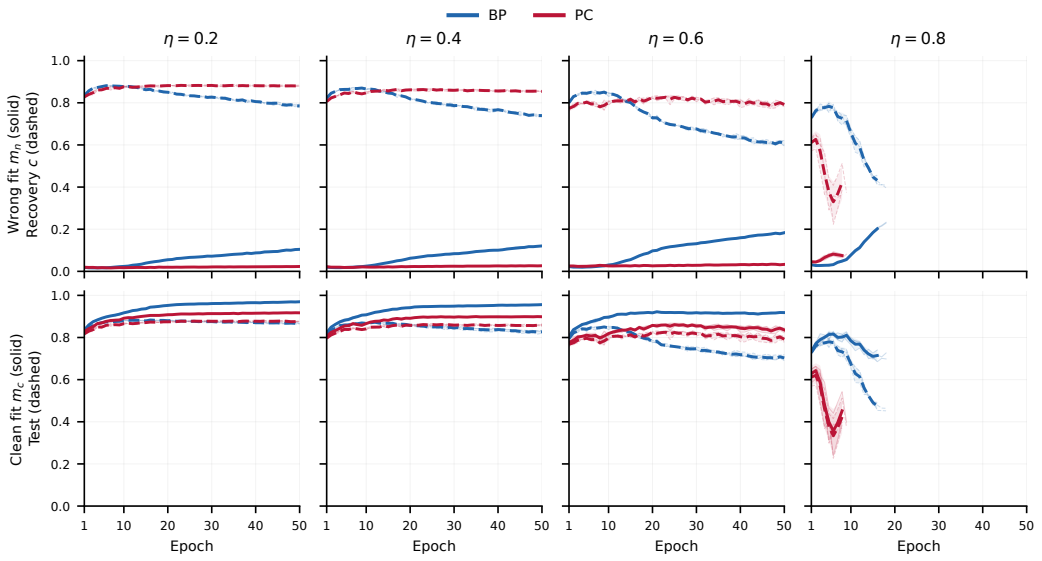


Figure 13: **Fashion-MNIST/MLP5 trajectories under symmetric label noise.** Columns give corruption rates; blue is BP and red PC. Solid/dashed lines show m_n/c above and m_c/test accuracy below. Bold lines and shading give three-seed means and sample SD over common recorded epochs; thin lines show individual runs through termination. PC uses squared error and BP cross-entropy; both use GELU. Test accuracy uses the full clean test set.

On CIFAR-100/VGG5, BP has not reached the late test-accuracy decline seen on CIFAR-10 and Fashion-MNIST: its test accuracy is still rising or stays near its maximum at the training endpoint, as shown in Figure 14. Under heavier label corruption, PC instead loses clean fitting and test accuracy after an early rise; its wrong-label fitting stays near chance. Both methods use squared error; PC uses leaky ReLU and BP uses GELU.

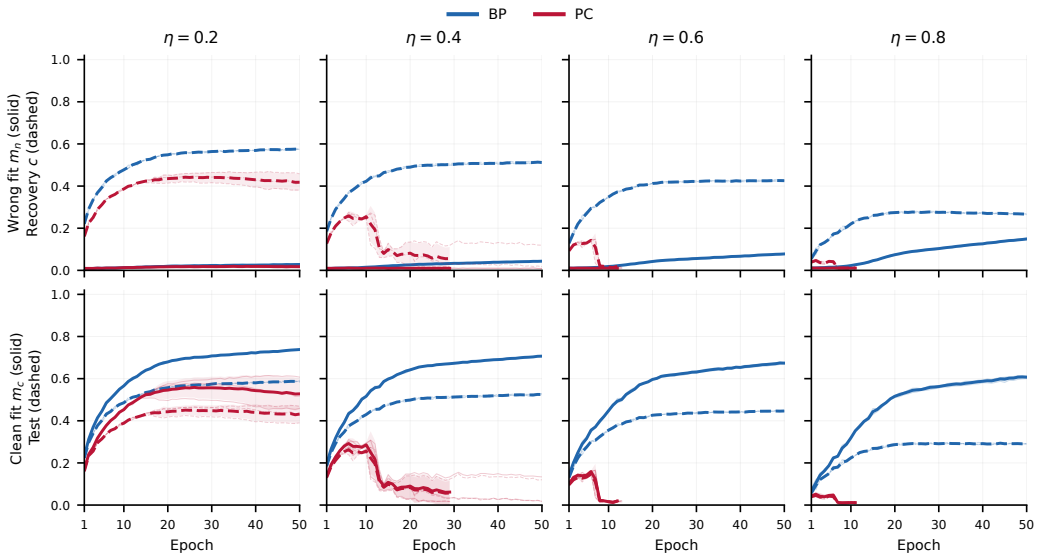


Figure 14: **CIFAR-100/VGG5 trajectories under symmetric label noise.** Columns, colors, line styles, and aggregation follow Figure 13. Both methods use squared error; PC uses leaky ReLU and BP GELU. PC runs at the two highest rates, and one at the next lower rate, end after recorded divergence. Test accuracy uses the full clean test set.

C PC/PC+F, INTERMEDIATE RULES, AND PAIRED UPDATES

The following measurements support the rule interventions in Section 4.

C.1 TARGET-CONDITIONED ABSORPTION ACROSS LAYERS

The absorption probe uses the same corrupted images under two target conditions. It records the displacement and residual norms before forming their ratio, and records complete-gradient norms for both weight rules. Figure 8 shows the forward mismatch, inferred residual, and native-target contrast; Table 8 gives the per-image state/residual and complete-gradient ratios. Figure 15 shows medians and interquartile ranges across layers.

The largest separation occurs at the output. Earlier layers retain similar state/residual scales and similar complete-gradient norms under the two rules. At the output, the large state/residual contrast accompanies a much smaller PC gradient. This localizes the transfer effect seen in the label-replacement analysis.

Table 8 extends the paired comparison in Table 1 across checkpoints and training rules. At PC-trained checkpoints, the wrong target produces a larger forward mismatch while both targets leave small residuals; the state/residual contrast increases as training proceeds. At PC+F-trained checkpoints, most of the forward mismatch remains after inference. The readout receives the target as its clamped state, while hidden-state motion changes the prediction entering it. The contrast in this table therefore locates accommodation in that inference pathway.

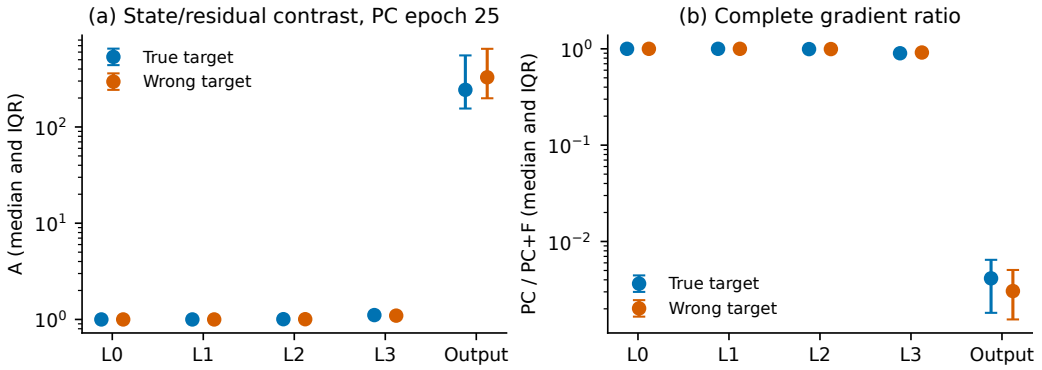


Figure 15: **Layerwise state/residual contrast and gradient ratio.** (a) State/residual contrast; (b) complete PC/PC+F gradient-norm ratio after shared inference. Points and bars show medians and interquartile ranges across paired images.

Training checkpoint	Target	Mean $\|d_L\|$	Mean $\|\epsilon_L\|$	Mean A	Mean G
PC, epoch 2	True	0.83683	0.026102	34.87	0.032696
PC, epoch 2	Wrong	1.03729	0.034283	34.46	0.033690
PC, epoch 25	True	0.76915	0.003454	474.76	0.004834
PC, epoch 25	Wrong	0.97871	0.003454	546.06	0.003675
PC, epoch 50	True	0.81636	0.000745	1271.73	0.000974
PC, epoch 50	Wrong	0.94474	0.000760	1508.33	0.000801
PC+F, epoch 2	True	0.79468	0.768612	1.03	0.968901
PC+F, epoch 2	Wrong	1.01442	0.981009	1.03	0.965162
PC+F, epoch 25	True	0.75694	0.713966	1.06	0.943896
PC+F, epoch 25	Wrong	0.93493	0.882803	1.06	0.943975
PC+F, epoch 50	True	0.83897	0.766954	1.09	0.914601
PC+F, epoch 50	Wrong	0.84924	0.777364	1.09	0.915196

Table 8: Output norms and transfer ratios for the paired corrupted-image pool underlying Figure 8. The first column identifies the rule used to train the checkpoint. At each frozen checkpoint, both gradients are evaluated after the same target-conditioned inference; $G = \|g_L^{\mathrm{PC}}\|/\|g_L^{\mathrm{F}}\|$ in every row. $A = \|d_L\|/\|\epsilon_L\|$ and G are formed per image before averaging over 200 images from one training seed. Complete readout gradients include weights and bias.

C.1.1 NUMERICAL LABEL-REPLACEMENT RESPONSES

Table 9 gives the checkpoint values underlying Figure 9a–c. The mean output-response cosine changes from positive to negative while the response magnitude is attenuated; earlier blocks retain stronger responses, so the whole-network contrast differs from the readout contrast.

Training	Epoch	$\bar{\rho}_{\mathrm{out}}$	$\bar{\rho}_{\mathrm{all}}$	$\bar{C}^{\Delta}_{\mathrm{out}}$	$\bar{C}^{\Delta}_{\mathrm{all}}$
PC	2	0.0296	0.1665	$+0.910$	$+0.318$
PC	25	0.000661	0.0825	-0.469	$+0.086$
PC	50	0.000799	0.0659	-0.410	$+0.075$
PC+F	25	0.944	0.944	$+1.000$	$+1.000$

Table 9: **Label-replacement transfer changes with layer and training state.** Entries are means of per-sample quantities, not ratios or cosines of batch means. All parameters include weights and biases. One training seed, 200 paired images.

C.2 INTERMEDIATE-RULE TRAINING

The interpolation campaign tests whether intermediate settings change the balance of clean fitting and noise memorization. It uses VGG5/CIFAR-10, symmetric corruption 0.4, and three seeds per setting. Its weight-learning-rate schedule is separate from the central benchmark.

Under shared hyperparameters, the successfully trained interior settings have closely grouped endpoint metrics, between the recorded PC and PC+F endpoints. The failed setting is excluded from the connected curve in Figure 6a. Thus the sweep contains both an intermediate learning regime and a training failure; it does not provide a strictly monotone ordering of every coefficient.

The accuracy-based search at the intermediate coefficient explores its own hyperparameters. Figure 6b plots the search outcomes, and panel (c) compares the selected intermediate run with separately tuned PC and the shared-sweep PC+F endpoint. The resulting configuration combines lower wrong-label fitting and higher test accuracy than that PC+F endpoint. These results make the training configuration part of the interpretation of an intermediate rule. The reported coefficient matches the recorded sweep; Eq. (7) specifies its mixed-input update.

D TARGET ADAPTATION, DIRECTION PROBES, AND HIDDEN CLAMPS

This appendix gives the protocols and detailed results for Section 5.

D.1 CONTROLLED TARGET SCALING AND DIRECTION

D.1.1 PROTOCOL AND RECONSTRUCTED STATISTICS

The output-target probe fixes a checkpoint and a corrupted image, stores its forward state, and performs fresh inference for each imposed target. Its label directions are the forward mismatches $v_c = d_L(y^c)$ and $v_n = d_L(y^n)$. The random direction v_r is an independent Gaussian direction rescaled to the norm of v_n. Targets are $y_a = h_L^0 + av$; the output displacement is therefore av. The random and wrong-target directions have equal norms within an image, while the true-target direction retains its own norm.

The delivered records contain displacement and residual norms for each layer, image, direction, and dose. Table 10 is recomputed from those records. Slopes are fitted within each image using positive doses and then summarized across images. At unit dose, the mean residual fraction is formed after division within each image. The relative gain range is $(\max_a \pi_i(a) - \min_a \pi_i(a))/\operatorname{mean}_a \pi_i(a)$; the table reports its maximum across images.

The slopes center near proportional scaling, with greater per-image deviations at the late PC checkpoint. Residual fractions differ substantially between the PC and PC+F training families. Reading the slope and gain together separates how the response changes with magnitude from how strongly

Weights	Target	Epoch	Slope mean ± SD	Mean π_L	Max. gain range
PC	Wrong	2	1.0000 ± 0.0000	0.03320	0.0004
PC	True	2	1.0000 ± 0.0000	0.03223	0.0007
PC	Wrong	25	0.9996 ± 0.0024	0.00366	0.0850
PC	True	25	0.9999 ± 0.0020	0.00481	0.0977
PC	Wrong	50	0.9953 ± 0.0102	0.00080	0.2051
PC	True	50	0.9984 ± 0.0101	0.00097	0.1414
PC+F	Wrong	25	1.0000 ± 0.0000	0.94417	0.0000
PC+F	True	25	1.0000 ± 0.0000	0.94369	0.0000

Table 10: Output-target sweeps, 200 corrupted images per checkpoint and one training seed per family. Positive doses are $0.1, 0.25, 0.5, 1, 2, 3$; dose zero is the forward target control. SD describes variation across images. Gain range is dimensionless.

it is attenuated. Small residuals make precision checks particularly relevant to the late-checkpoint variation.

$$\bar{\mathbf{1}} = \frac{\mathbf{1}}{\sqrt{K}}, \qquad Q = I - \bar{\mathbf{1}}\bar{\mathbf{1}}^\top, \qquad v = Qv + (\bar{\mathbf{1}}^\top v)\bar{\mathbf{1}}. \tag{18}$$

One-hot label replacements satisfy $\bar{\mathbf{1}}^\top(y^n - y^c) = 0$. Each forward mismatch d_L has common component $\bar{\mathbf{1}}^\top d_L$ set by the forward output, whereas an unconstrained Gaussian direction samples this component freely. Separately varying the centered component and the common component distinguishes class-contrast geometry from output-offset geometry. The common-offset intervention in Figure 10b leaves a mean residual fraction of 0.98891 at unit dose. Its gain is nearly unchanged across the positive dose sweep. Centered drives and basis responses resolve the class-contrast response in Figure 10c–d and Appendix D.2.

Dose	Mean residual fraction	Mean complete-gradient ratio
0.1	0.003659	0.003659
1.0	0.003658	0.003675
3.0	0.003657	0.003830

Table 11: Residual and complete-gradient response at PC epoch 25, wrong-target channel, 200 paired images. Both columns are means of per-image ratios from the same dose-probe records. Complete gradients include weights and biases.

Table 11 shows a nearly unchanged residual fraction across these doses while the complete-gradient ratio increases. At a fixed target, Eq. (50) separates these measurements through the input norm factor. Thus radial proportionality of the residual does not imply a constant complete-gradient gain as the inferred input moves.

D.1.2 Probe initialization and boundary conditions

Every imposed target receives fresh inference from the saved forward snapshot, with unchanged weights and a reset state optimizer. The zero-dose target checks the forward baseline. Vector reconstruction, scaling, and addition are reported in Appendices D.2 and D.3.

The hidden-input experiment fixes both the output target and the selected hidden state at every inference step; other hidden states remain free.

D.2 Centered response maps and re-inferred label changes

D.2.1 Basis normalization and response spectra

For each image i, the coordinate probe drives each output coordinate j with amplitude s_i and saves the resulting output residual as column j of a matrix R_i. The response to a drive $s_i v$ is then reconstructed

as $R_i v$, and the normalized map $\widehat{R}_i = R_i / s_i$ separates the drive amplitude from the response gain. The matrix is an empirical finite-amplitude basis response;

Family	Seed/epoch	σ_1	σ_2	σ_{10}	$\|\widehat{R}_i Q\|_2$
Original	0/25	0.9889	0.00064	0.000038	0.00070
Shared	0/2	0.9195	0.06331	0.013404	0.06417
Shared	0/50	0.9999	0.00166	0.000088	0.00166
Shared	1/25	0.9773	0.00171	0.000073	0.00171
Shared	2/25	0.9843	0.00159	0.000062	0.00160

Table 12: Median normalized response gains over 60 images per checkpoint. The original epoch-25 checkpoint and the shared-configuration checkpoints are separate training families. Singular values are computed per image before summarization.

The shared seed-0 trajectory develops a sharper separation between the common mode and the centered modes during training. Later checkpoints from the other seeds also exhibit a dominant common mode. The small remaining singular values retain a nonzero response within the class-contrast subspace.

At the original epoch-25 checkpoint, the median relative reconstruction error for the positive common-offset probe is 1.25×10^{-5}. Centered probes have larger relative errors because their remaining residuals are much smaller (Table 13). Direct centered responses and basis singular values agree on the separation of gains.

Vector check	Median relative error	Maximum relative error
Common-offset reconstruction	1.25×10^{-5}	2.60×10^{-5}
Centered random reconstruction	1.69×10^{-2}	3.66×10^{-2}
Centered label reconstruction	1.58×10^{-2}	5.49×10^{-2}
Label-change gradient identity	1.19×10^{-7}	2.58×10^{-7}

Table 13: Independent reconstructions from the saved vectors at the original PC epoch-25 checkpoint. Each error is normalized by its own direct response.

D.2.2 Contributions to the label-change gradient

Equation (13) yields the exact norm identity for the readout weights

$$\|\Delta g_V\|_F^2 = \sum_{j=1}^{3} \|t_j\|_F^2 \tag{19}$$

The pairwise cosines in Figure 9d–e normalize the inner products by the contributing term norms. Small absolute inner products can arise from small terms and therefore do not by themselves identify orthogonality. The bias change has norm $\|\Delta\epsilon\|$; its nonzero response accompanies the weight response. For the forward-input rule, both target conditions retain the same input and the label-change response reduces to $\Delta g_V^{\mathrm{F}} = -(y^n - y^c)(h_{L-1}^0)^\top$.

D.3 Native targets and hidden-input interventions

D.3.1 Native observed targets

The native-target experiment samples clean and corrupted images separately from each training run's corruption mask. Each noisy target is reconstructed with that run's data seed and checked against its observed label before inference. Sampling balances true classes within each pool. PC and PC+F checkpoints from the same seed use identical probe IDs, and each noisy image receives an additional true-target inference. This directly separates the native C/N comparison from the paired N/N2C target intervention.

Table 14 reports the result. On PC-trained checkpoints, noisy targets have a larger state/residual contrast and a smaller complete readout-gradient ratio than clean targets in each seed. Within the

Training	Condition	C_{all}	C_{out}	A_{out}	G_{out}
PC	C	0.0664 ± 0.0157	-0.2056 ± 0.0277	360.57 ± 72.32	0.00628 ± 0.00008
PC	N	0.0750 ± 0.0091	-0.1259 ± 0.0503	547.25 ± 159.03	0.00368 ± 0.00043
PC	N2C	0.0686 ± 0.0120	-0.2131 ± 0.0331	425.58 ± 121.12	0.00494 ± 0.00068
PC+F	C	0.9998 ± 0.0000	0.9999 ± 0.0000	1.06 ± 0.00	0.94359 ± 0.00066
PC+F	N	0.9999 ± 0.0000	0.9999 ± 0.0000	1.06 ± 0.00	0.94359 ± 0.00112
PC+F	N2C	0.9998 ± 0.0000	0.9999 ± 0.0000	1.06 ± 0.00	0.94348 ± 0.00037

Table 14: Native-target probes at epoch 25: means and sample SD across three training-seed means. Each seed uses 120 clean and 120 corrupted images, balanced by true class. N2C re-infers the N images under true targets. Cosines compare rules under one target; they are distinct from label-replacement cosines.

noisy pool, replacing the observed target by the true target increases the gradient ratio and reduces the absorption contrast. The whole-network rule cosine remains small in both pools, while the readout cosine exhibits a larger condition contrast. PC+F-trained checkpoints instead retain closely aligned rule gradients across the target conditions. At those weights, the same inferred-input rule produces little readout absorption: the filtering observed at PC-trained checkpoints depends on the learned weights as well as the rule.

D.3.2 The last-hidden clamp follows the affine readout

Clamp the last hidden state to $h_{L-1}(a) = h^0_{L-1} + a\, d_{L-1}$, a fraction a of the displacement produced by ordinary target-conditioned inference, while retaining the output target. With the affine readout $Vh_{L-1} + b$, the forward mismatch d_L and the ordinary residual ϵ_L of Eqs. (2)–(3),

$$\epsilon(a) = y - V[h^0_{L-1} + a\, d_{L-1}] - b \tag{20}$$
$$= d_L - aVd_{L-1} \tag{21}$$
$$= (1-a)d_L + a\epsilon_L, \tag{22}$$
$$\|\epsilon(a)\|^2 = (1-a)^2\|d_L\|^2 + a^2\|\epsilon_L\|^2 + 2a(1-a)\langle d_L, \epsilon_L\rangle. \tag{23}$$

The saved norms determine the cross inner product from the half-dose response. Substituting it predicts the quarter-dose and double-dose responses with maximum relative norm error 4.44×10^{-7}. The observed residual curve is thus the expected consequence of moving the readout input along this affine path.

Clamping the preceding hidden layer leaves the last-hidden state free to respond to the target. Its output residual remains small across the recorded doses. The imposed state norms are plotted separately in Figure 16b: the same relative dose corresponds to different absolute displacements at the two layers.

D.3.3 Linearity on the weak-response scale

At the frozen checkpoint, let $R(v) = \epsilon_L(v) - \epsilon_L(0)$ be the change in output residual caused by an output drive v. For centered drives v, v' ($Qv = v$, $Qv' = v'$) the probes measure

$$\text{err}_{\text{scale}} = \frac{\|R(2v) - 2R(v)\|}{\|R(v)\|}, \qquad \text{err}_{\text{sign}} = \frac{\|R(v) + R(-v)\|}{\|R(v)\|}, \tag{24}$$

$$\text{err}_{\text{add}} = \frac{\|R(v+v') - R(v) - R(v')\|}{\|R(v) + R(v')\|}. \tag{25}$$

The measured errors are small but resolved on the centered-response scale (Table 15). These probes extend the basis reconstruction with direct tests of scaling, sign reversal, and addition.

For a mixed drive that adds a centered drive av to a common shift $s\bar{\mathbf{1}}$, write $R_{\text{mix}} = R(s\bar{\mathbf{1}}+av)$, $R_{\text{c}} = R(s\bar{\mathbf{1}})$ and $R_\perp = aR(v)$; the superposition error is $\zeta = R_{\text{mix}} - R_{\text{c}} - R_\perp$, and $\chi = \|\zeta\|/\|R_{\text{mix}}\|$ is

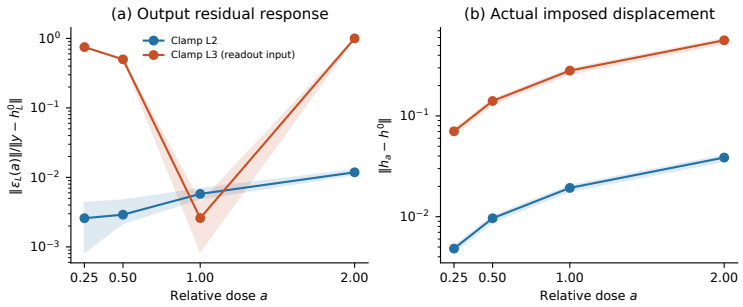


Figure 16: Hidden-state clamp responses at the readout. (a) Output residual fraction; (b) imposed hidden-state displacement. Lines and bands show medians and interquartile ranges.

its size relative to the full response. Triangle inequalities give

$$\|R_{\text{mix}}\| \geq \|R_{\text{c}} + R_{\perp}\| - \|\zeta\|, \qquad \|R_{\text{mix}}\| \leq \|R_{\text{c}} + R_{\perp}\| + \|\zeta\|, \tag{26}$$

$$\frac{\chi\,|\|R_{\text{c}}\| - \|R_{\perp}\||}{(1+\chi)\|R_{\perp}\|} \leq \frac{\|\zeta\|}{\|R_{\perp}\|} \leq \frac{\chi(\|R_{\text{c}}\| + \|R_{\perp}\|)}{(1-\chi)\|R_{\perp}\|}, \qquad \chi < 1. \tag{27}$$

These bounds relate full-response normalization to the centered contribution. The scalar records bound the discrepancy relative to the centered contribution (Table 15). These bounds evaluate approximate superposition on the weak-response scale as a common shift is combined with a centered perturbation. Within a few percent of the centered response, the map is linear in scale, sign and addition, also when a common shift is present. This is the property that licenses reconstructing responses to arbitrary drives from coordinate probes in Appendix D.2.

Check	Median relative error	Maximum / upper bound
Centered scaling	0.02196	0.05676
Centered additivity	0.01233	0.03760
Centered sign reversal	0.02345	0.06286
Mixed drive, $a = 0.25$	[0.029528, 0.029534]	0.06351
Mixed drive, $a = 0.5$	[0.014384, 0.014390]	0.03455
Mixed drive, $a = 1.0$	[0.007275, 0.007280]	0.01442

Table 15: Weak-response linearity at the original PC epoch-25 checkpoint, 32 images. Mixed-drive errors are normalized by the scaled centered response and bounded using Eq. (27); the middle column gives bounds on the median. Other rows report direct errors with the normalizations defined above.

D.4 Readout residual under longer inference at fixed weights

Proposition 2 distinguishes the finite-step operator P_T from its equilibrium limit. This experiment asks which of the two describes the readout signal that PC learns from, by extending inference at frozen weights beyond the training budget. The instrumented sweep uses frozen weights, forward initialization, a clamped output, and fresh heavy-ball momentum for each image and target. It extends inference to 80 updates. The complete runs cover 16 images per condition at ten checkpoints: both training families at epochs 2, 25, and 50 for seed 0, plus epoch 25 for seeds 1 and 2. At the training budget, the output displacement, residual, and both gradient norms reproduce the separate paired probe up to floating-point differences.

Table 16 shows that the readout residual is smallest near the training budget and grows again as inference continues, while the total energy keeps falling. Longer inference redistributes residual across layers, and part of it returns to the readout; the free-state gradient is still nonzero at the end of the sweep. The finite-step operator of Proposition 2 allows this pattern: gradient descent with $0 < \eta_h \mu_j < 1$ decreases the stacked residual of all layers, and hence the total energy, monotonically, but its output block $O(I - P_T)r_0$ need not decrease monotonically. Early in inference the layers next to the output absorb the mismatch before it has spread to lower layers, so the output residual falls

Inference updates	Total energy	Residual fraction	Complete-gradient ratio
0	0.45903	1.00000	1.00000
7	0.02757	0.00372	0.00375
10	0.02408	0.01850	0.01861
20	0.01588	0.04697	0.04725
40	0.01278	0.03111	0.03130
80	0.01126	0.02536	0.02552

Table 16: Medians over 16 wrong-target images at the seed-0 PC epoch-25 checkpoint. The gradient ratio includes readout weights and bias. Inference duration changes at fixed weights; these rows are distinct from training separate models at different inference budgets.

below the value it takes once the residual is shared across the network. Absorption at the output is therefore a property of the finite budget used in training rather than of the equilibrium. This also gives one reading of the inference-duration runs in Section 3.3: changing the budget changes which readout residual the weights learn from, rather than adding fitting capacity.

E THEORY AND STEP-BY-STEP DERIVATIONS

The derivations connect the local gradient, finite state dynamics, and the target probes.

E.1 DETAILED DERIVATIONS OF STATE-TO-WEIGHT TRANSFER

This appendix develops the main results from the local learning rule. In readout-only derivations below, h abbreviates the last hidden input h_{L-1}, never the clamped output h_L. We first separate the effect of the input on the residual from its effect on the parameter Jacobian. We then solve the state dynamics in a fixed local model and in an anchored quadratic readout. The resulting expressions connect the state response, the remaining residual, and the complete weight gradient.

E.1.1 EXACT READOUT DECOMPOSITION

Fix a layer l. During weight differentiation the input states h_{l-1}^T, h_{l-1}^0 and the target state h_l^T are stored constants. The residual and the displacement of Eqs. (2)–(3) are

$$\epsilon_l = h_l^T - f_l(h_{l-1}^T; W_l), \qquad d_l = h_l^T - f_l(h_{l-1}^0; W_l). \tag{28}$$

Subtracting them cancels the common target state:

$$\begin{aligned} d_l - \epsilon_l &= [h_l^T - f_l(h_{l-1}^0; W_l)] - [h_l^T - f_l(h_{l-1}^T; W_l)] && (29)\\ &= f_l(h_{l-1}^T; W_l) - f_l(h_{l-1}^0; W_l) && (30)\\ &=: q_l. && (31) \end{aligned}$$

Hence $d_l = \epsilon_l + q_l$, proving the exact part of Proposition 1. For the local transport expression, apply the fundamental theorem of calculus along the segment from h_{l-1}^0 to $h_{l-1}^T = h_{l-1}^0 + d_{l-1}$, assuming a continuous input Jacobian $J_{h,l}$ on that segment:

$$\begin{aligned} q_l &= f_l(h_{l-1}^0 + d_{l-1}; W_l) - f_l(h_{l-1}^0; W_l) && (32)\\ &= \int_0^1 J_{h,l}(h_{l-1}^0 + s\, d_{l-1})\, d_{l-1}\, \mathrm{d}s && (33)\\ &= J_{h,l}(h_{l-1}^0)\, d_{l-1} + \mathrm{rem}(d_{l-1}), && (34)\\ \mathrm{rem}(d_{l-1}) &= \int_0^1 [J_{h,l}(h_{l-1}^0 + s\, d_{l-1}) - J_{h,l}(h_{l-1}^0)]\, d_{l-1}\, \mathrm{d}s. && (35) \end{aligned}$$

Differentiability gives $\mathrm{rem}(d_{l-1}) = o(\|d_{l-1}\|)$. If the Jacobian is Lipschitz on this segment with constant Λ, the integral also gives

$$\|\mathrm{rem}(d_{l-1})\| \le \int_0^1 \Lambda s \|d_{l-1}\|^2 \,\mathrm{d}s \tag{36}$$

$$= \frac{\Lambda}{2}\|d_{l-1}\|^2. \tag{37}$$

For a linear layer, the Jacobian is constant and the remainder vanishes. Thus latent motion is transported exactly through the readout matrix and locally through a differentiable nonlinear layer.

E.1.2 The input changes the complete weight gradient

PC and PC+F differentiate the local energy terms

$$\mathcal{E}_l^{\mathrm{PC}}(W_l) = \tfrac{1}{2}\|h_l^T - f_l(h_{l-1}^T; W_l)\|^2, \qquad \mathcal{E}_l^{\mathrm{F}}(W_l) = \tfrac{1}{2}\|h_l^T - f_l(h_{l-1}^0; W_l)\|^2, \tag{38}$$

whose prediction Jacobians are $J_{W,l}^T$ and $J_{W,l}^0$. Holding the stored states fixed gives

$$g_l^{\mathrm{PC}} = -(J_{W,l}^T)^\top \epsilon_l, \tag{39}$$

$$g_l^{\mathrm{F}} = -(J_{W,l}^0)^\top d_l = -(J_{W,l}^0)^\top(\epsilon_l + q_l). \tag{40}$$

Their difference separates into two contributions:

$$g_l^{\mathrm{F}} - g_l^{\mathrm{PC}} = -(J_{W,l}^0)^\top(\epsilon_l + q_l) + (J_{W,l}^T)^\top \epsilon_l \tag{41}$$

$$= \underbrace{-(J_{W,l}^0)^\top q_l}_{\text{prediction-change contribution}} + \underbrace{(J_{W,l}^T - J_{W,l}^0)^\top \epsilon_l}_{\text{input-Jacobian contribution}}. \tag{42}$$

The fixed-Jacobian local model retains the first contribution. The complete learning rule contains both because its parameter derivative is evaluated at the selected input.

For the linear readout Vh_{L-1}, the parameter differential is $\mathrm{d}(Vh_{L-1}^T) = (\mathrm{d}V)h_{L-1}^T$. Using the Frobenius inner product,

$$\mathrm{d}\mathcal{E}_L^{\mathrm{PC}} = -\epsilon_L^\top (\mathrm{d}V) h_{L-1}^T \tag{43}$$

$$= \mathrm{tr}\left[(-\epsilon_L (h_{L-1}^T)^\top)^\top \mathrm{d}V\right], \tag{44}$$

$$\nabla_V \mathcal{E}_L^{\mathrm{PC}} = -\epsilon_L (h_{L-1}^T)^\top. \tag{45}$$

Consequently, the input-choice difference becomes

$$g_V^{\mathrm{F}} - g_V^{\mathrm{PC}} = -(\epsilon_L + q_L)(h_{L-1}^0)^\top + \epsilon_L (h_{L-1}^T)^\top \tag{46}$$

$$= -q_L (h_{L-1}^0)^\top + \epsilon_L \, d_{L-1}^\top. \tag{47}$$

This is the linear-layer form of Eq. (42). It explains why a comparison of residuals and a comparison of complete gradients measure different stages of the transfer pathway.

For an affine readout $Vh + b$, include the bias through $\widetilde{h} = (h^\top, 1)^\top$ and $\widetilde{V} = [V\ b]$. The complete weight-and-bias gradient is $g_{\widetilde{V}} = -\epsilon_L \widetilde{h}^\top$. The Frobenius norm of an outer product factorizes, giving, for each fixed target and $\|d_L\| > 0$,

$$\|g_{\widetilde{V}}^{\mathrm{PC}}\|_F = \|\epsilon_L\| \, \|\widetilde{h}^T\|, \tag{48}$$

$$\|g_{\widetilde{V}}^{\mathrm{F}}\|_F = \|d_L\| \, \|\widetilde{h}^0\|, \tag{49}$$

$$G(y) = \frac{\|\epsilon_L(y)\|}{\|d_L(y)\|} \frac{\|\widetilde{h}^T(y)\|}{\|\widetilde{h}^0\|}. \tag{50}$$

Thus the residual fraction and the input-norm ratio jointly determine the single-target gradient ratio at the readout. Label-replacement gradients subtract two such outer products; their exact expansion is derived below.

E.1.3 DIFFERENTIATING THE TARGET RESPONSE

Proposition 3 (Target response of the complete gradient)**.** *Fix V, b and the target-independent forward input $h^0 = h^0_{L-1}$. For a differentiable inference map $h(y)$, set $S_T = D_y h(y)$ and $\epsilon = y - Vh(y) - b$. For a target perturbation in direction v,*

$$D_y g_V^{\mathrm{PC}}[v] = -[(I - VS_T)v]h(y)^\top - \epsilon(S_T v)^\top, \qquad (51)$$
$$D_y g_V^{\mathrm{F}}[v] = -v(h^0)^\top.$$

To prove the proposition, keep V and any readout bias fixed, and write $h = h^T_{L-1}(y)$. The displayed derivation sets the fixed bias to zero; including it in the residual leaves the derivative unchanged. The target differential of the state is $\mathrm{d}h = S_T\,\mathrm{d}y$. Since $\epsilon = y - Vh$, its differential is

$$\mathrm{d}\epsilon = \mathrm{d}y - V\,\mathrm{d}h \qquad (52)$$
$$= (I - VS_T)\,\mathrm{d}y. \qquad (53)$$

Applying the product rule to the complete gradient gives

$$\mathrm{d}g_V^{\mathrm{PC}} = \mathrm{d}(-\epsilon h^\top) \qquad (54)$$
$$= -(\mathrm{d}\epsilon)h^\top - \epsilon(\mathrm{d}h)^\top \qquad (55)$$
$$= -[(I - VS_T)\mathrm{d}y]h^\top - \epsilon(S_T\mathrm{d}y)^\top. \qquad (56)$$

Choosing $\mathrm{d}y = v$ proves the PC expression. The initial input h^0 is independent of the target, so the forward-input rule gives

$$\mathrm{d}g_V^{\mathrm{F}} = -\mathrm{d}(y - Vh^0)(h^0)^\top \qquad (57)$$
$$= -(\mathrm{d}y)(h^0)^\top. \qquad (58)$$

These derivatives keep the inference response to y while treating the stored states as fixed in the separate weight derivative that defines each learning rule.

E.1.4 FINITE LABEL REPLACEMENTS

The empirical probe replaces the true target y^c by the wrong target y^n. With the notation of Eq. (13),

$$\Delta y = y^n - y^c, \quad h_c = h^T_{L-1}(y^c), \quad \Delta h = h^T_{L-1}(y^n) - h_c, \quad \Delta\epsilon = \epsilon_L(y^n) - \epsilon_c = \Delta y - V\Delta h, \qquad (59)$$

where $\epsilon_c = \epsilon_L(y^c)$. Expanding the two endpoint gradients yields

$$\Delta g_V^{\mathrm{PC}} = -(\epsilon_c + \Delta\epsilon)(h_c + \Delta h)^\top + \epsilon_c h_c^\top \qquad (60)$$
$$= \underbrace{-\Delta\epsilon\, h_c^\top}_{t_1}\ \underbrace{-\epsilon_c \Delta h^\top}_{t_2}\ \underbrace{-\Delta\epsilon\,\Delta h^\top}_{t_3}. \qquad (61)$$

The forward input is common to the two targets, giving

$$\Delta g_V^{\mathrm{F}} = -(y^n - Vh^0_{L-1})(h^0_{L-1})^\top + (y^c - Vh^0_{L-1})(h^0_{L-1})^\top \qquad (62)$$
$$= -\Delta y\,(h^0_{L-1})^\top. \qquad (63)$$

For an affine layer, the bias-gradient differences are respectively $-\Delta\epsilon$ and $-\Delta y$. The combined magnitude of the three terms is

$$\|\Delta g_V^{\mathrm{PC}}\|_F^2 = \sum_{j=1}^{3} \|t_j\|_F^2 + 2\sum_{j<k} \langle t_j, t_k\rangle_F. \qquad (64)$$

The cross inner products record reinforcement and cancellation between the residual and input contributions. Because the Frobenius norm of an outer product factorizes, the expansion also gives the bound

$$\|\Delta g_V^{\mathrm{PC}}\|_F \le \|\Delta\epsilon\|\|h_c\| + \|\epsilon_c\|\|\Delta h\| + \|\Delta\epsilon\|\|\Delta h\|. \qquad (65)$$

The exact expansion is used for finite replacements; its first-order terms recover the differential expression as the target change tends to zero.

E.1.5 SOLVING THE FIXED LOCAL RESIDUAL MODEL

We now prove Proposition 2. For fixed M_z, M_w, hold $w = 0$ and consider

$$\mathcal{E}(z) = \tfrac{1}{2}\|r_0 - M_z z\|^2, \qquad H = M_z^\top M_z. \tag{66}$$

The gradient and the state recurrence are

$$\nabla_z \mathcal{E}(z) = -M_z^\top (r_0 - M_z z) = Hz - M_z^\top r_0, \tag{67}$$

$$z_{t+1} = (I - \eta_h H) z_t + \eta_h M_z^\top r_0. \tag{68}$$

Starting from $z_0 = 0$, successive substitution gives

$$z_1 = \eta_h M_z^\top r_0, \tag{69}$$

$$z_2 = \eta_h [(I - \eta_h H) + I] M_z^\top r_0, \tag{70}$$

$$z_T = \eta_h \sum_{k=0}^{T-1} (I - \eta_h H)^k M_z^\top r_0. \tag{71}$$

The forcing vector belongs to $\mathrm{range}(M_z^\top) = \mathrm{range}(H)$. On an eigenvector of H with eigenvalue $\mu > 0$, the geometric sum evaluates to

$$\eta_h \sum_{k=0}^{T-1} (1 - \eta_h \mu)^k = \frac{1 - (1 - \eta_h \mu)^T}{\mu}. \tag{72}$$

The nullspace receives no forcing and remains at its zero initialization. Combining the active eigenspaces gives

$$z_T = H^\dagger [I - (I - \eta_h H)^T] M_z^\top r_0. \tag{73}$$

Substituting into the residual,

$$r_T = r_0 - M_z z_T \tag{74}$$

$$= \left\{ I - M_z H^\dagger [I - (I - \eta_h H)^T] M_z^\top \right\} r_0 \tag{75}$$

$$= (I - P_T) r_0. \tag{76}$$

Finally, differentiating the local energy with respect to w at fixed z_T gives

$$\nabla_w \mathcal{E}(z_T, w) = -M_w^\top (r_0 - M_z z_T - M_w w), \tag{77}$$

$$g_w = \nabla_w \mathcal{E}(z_T, w)|_{w=0} = -M_w^\top (I - P_T) r_0. \tag{78}$$

This completes the finite-step result.

E.1.6 SPECTRAL ABSORPTION AND THE EQUILIBRIUM DECOMPOSITION

Take a thin singular value decomposition $M_z = \Phi \Sigma \Psi^\top$ of rank r, with positive singular values σ_j and $\mu_j = \sigma_j^2$. The columns ϕ_j of Φ span the residual directions that state motion can reach, and the columns of Ψ span the active state directions. Then

$$H = \Psi \Sigma^2 \Psi^\top, \qquad H^\dagger = \Psi \Sigma^{-2} \Psi^\top, \tag{79}$$

$$P_T = \Phi \Sigma \Psi^\top \Psi \Sigma^{-2} \Psi^\top [I - (I - \eta_h H)^T] \Psi \Sigma \Phi^\top \tag{80}$$

$$= \Phi \, \mathrm{diag}\big(1 - (1 - \eta_h \mu_j)^T\big) \, \Phi^\top. \tag{81}$$

Thus $P_T^\top = P_T$ at every finite step, its eigenvectors ϕ_j lie in residual space, and P_T vanishes on $\mathrm{col}(M_z)^\perp$. Write the initial residual as $r_0 = \sum_j \langle \phi_j, r_0 \rangle \phi_j + r_\perp$ with $\Phi^\top r_\perp = 0$. The state-explained component and the remaining residual are

$$P_T r_0 = \sum_j \alpha_j(T) \langle \phi_j, r_0 \rangle \phi_j, \qquad \alpha_j(T) = 1 - (1 - \eta_h \mu_j)^T, \tag{82}$$

$$r_T = \sum_j [1 - \alpha_j(T)] \langle \phi_j, r_0 \rangle \phi_j + r_\perp. \tag{83}$$

When $0 < \eta_h \mu_j < 2$, the active residual factors tend to zero, giving

$$P_\infty = \Phi\Phi^\top = M_z H^\dagger M_z^\top, \tag{84}$$

$$r_0 = \underbrace{\Phi\Phi^\top r_0}_{\text{state-explainable component}} + \underbrace{(I - \Phi\Phi^\top) r_0}_{\text{orthogonal residual}}. \tag{85}$$

The limiting operator is symmetric and idempotent. At finite T its eigenvalues are $\alpha_j(T)$. For a small accumulated modal step, the binomial expansion gives

$$\alpha_j(T) = T\eta_h\mu_j - \binom{T}{2}(\eta_h\mu_j)^2 + \cdots \tag{86}$$

$$\simeq T\eta_h\mu_j \qquad \text{when } T\eta_h\mu_j \ll 1, \tag{87}$$

which displays the joint role of inference time and curvature in the early response. The parameter response $M_w^\top (I - P_T) r_0$ weights residual directions by their effect on the parameters, so attenuation of the residual and attenuation of the weight gradient are separate quantities in the model.

E.1.7 Magnitude scaling and residual-space directions

For a drive $r_0 = av$ with dose $a > 0$ and $\|v\| = 1$, linearity of the fixed local recurrence gives

$$r_T(av) = (I - P_T)(av) = a(I - P_T)v, \tag{88}$$

$$\frac{\|r_T(av)\|}{\|av\|} = \|(I - P_T)v\|. \tag{89}$$

Writing $v = \sum_j \langle \phi_j, v\rangle \phi_j + v_\perp$,

$$P_T v = \sum_j \left[1 - (1 - \eta_h\mu_j)^T\right] \langle \phi_j, v\rangle \phi_j, \tag{90}$$

$$(I - P_T)v = \sum_j (1 - \eta_h\mu_j)^T \langle \phi_j, v\rangle \phi_j + v_\perp. \tag{91}$$

The components are orthogonal, so taking the squared norm proves Eq. (121). The perpendicular term records the part of the drive outside the range accessible to state motion. For an output-only measurement, let $O^\top$ embed an output vector into the stacked residual ($OO^\top = I$) and impose $y(a) = h_L^0 + av$ with $\|v\| = 1$, so that $r_0 = aO^\top v$. The output clamp gives

$$d_L(a) = y(a) - h_L^0 = av, \tag{92}$$

$$\frac{\|\epsilon_L(a)\|}{\|d_L(a)\|} = \|O(I - P_T)O^\top v\|, \qquad a > 0. \tag{93}$$

The selected components need not retain the orthogonality used above. At a free layer, $d_l(a) = h_l^T(a) - h_l^0$ is instead determined by the inference trajectory; its norm must be evaluated from that trajectory.

A vector-valued linear response satisfies both $r_T(av) = a r_T(v)$ and $r_T(v + v') = r_T(v) + r_T(v')$. A norm sweep measures radial norm scaling; vector records test whether the response direction is also preserved under scaling, and paired summed drives test additivity. For an affine response these comparisons apply to $r_T(v) - r_T(0)$.

E.1.8 Anchored quadratic inference and target sensitivity

Proposition 4 (Anchored quadratic inference). *For $\beta > 0$, fixed V, h^0, and $\mathcal{E}(h) = \frac{1}{2}\|h - h^0\|^2 + \frac{\beta}{2}\|y - Vh\|^2$, let $H_\beta = I + \beta V^\top V$. The equilibrium is $h^* = H_\beta^{-1}(h^0 + \beta V^\top y)$. Ordinary gradient descent from h^0 gives $h^T - h^* = (I - \eta_h H_\beta)^T (h^0 - h^*)$, and*

$$y - Vh^* = (I + \beta VV^\top)^{-1} d_L, \qquad d_L = y - Vh^0. \tag{94}$$

In this state-inference model, β weights the inference objective. The complete readout gradients in Proposition 3 use the unit-weight local output loss. For Proposition 4, expand the state-energy gradient:

$$\nabla_h \mathcal{E}(h) = (h - h^0) - \beta V^\top (y - Vh) \tag{95}$$
$$= (I + \beta V^\top V)h - h^0 - \beta V^\top y \tag{96}$$
$$= H_\beta h - h^0 - \beta V^\top y. \tag{97}$$

Positive definiteness follows from

$$u^\top H_\beta u = \|u\|^2 + \beta \|Vu\|^2 > 0 \qquad (u \neq 0). \tag{98}$$

Setting the gradient to zero gives the unique solution

$$h^* = H_\beta^{-1}(h^0 + \beta V^\top y). \tag{99}$$

The ordinary gradient-descent recurrence is

$$h^{t+1} = (I - \eta_h H_\beta)h^t + \eta_h(h^0 + \beta V^\top y). \tag{100}$$

Subtracting the stationary equation and iterating gives

$$h^{t+1} - h^* = (I - \eta_h H_\beta)(h^t - h^*), \tag{101}$$
$$h^T - h^* = (I - \eta_h H_\beta)^T (h^0 - h^*), \tag{102}$$
$$h^T = (I - \eta_h H_\beta)^T h^0 + [I - (I - \eta_h H_\beta)^T]h^*. \tag{103}$$

Because h^0 is independent of the target,

$$\frac{\partial h^*}{\partial y} = H_\beta^{-1} \beta V^\top, \tag{104}$$
$$S_T = [I - (I - \eta_h H_\beta)^T] H_\beta^{-1} \beta V^\top \tag{105}$$
$$= H_\beta^{-1}[I - (I - \eta_h H_\beta)^T] \beta V^\top. \tag{106}$$

The final equality uses commutation of functions of H_β.

To express the equilibrium output residual directly, let $\epsilon^* = y - Vh^*$. Stationarity gives

$$h^* - h^0 = \beta V^\top \epsilon^*, \tag{107}$$
$$Vh^* - Vh^0 = \beta V V^\top \epsilon^*, \tag{108}$$
$$d_L = \epsilon^* + \beta V V^\top \epsilon^*, \tag{109}$$
$$\epsilon^* = (I + \beta V V^\top)^{-1} d_L. \tag{110}$$

This completes the proof. The equilibrium filter acts in output space, while Eq. (103) describes the finite-time hidden-state trajectory from which the full gradient is formed.

E.1.9 Momentum response and stability

For the heavy-ball convention with velocity ν_t,

$$\nu_{t+1} = m\nu_t - \eta_h \nabla \mathcal{E}(h_t), \qquad h_{t+1} = h_t + \nu_{t+1}, \qquad \nu_0 = 0, \tag{111}$$

use $\nu_t = h_t - h_{t-1}$ to eliminate the velocity:

$$h_{t+1} = (1 + m)h_t - m h_{t-1} - \eta_h \nabla \mathcal{E}(h_t). \tag{112}$$

Along an eigenvector of the quadratic Hessian with eigenvalue μ, let ξ_t be the distance from equilibrium. Then

$$\xi_{t+1} = \gamma\, \xi_t - m\, \xi_{t-1}, \qquad \gamma = 1 + m - \eta_h \mu, \qquad \xi_1 = (1 - \eta_h \mu)\xi_0. \tag{113}$$

The characteristic roots are

$$s_\pm = \frac{\gamma \pm \sqrt{\gamma^2 - 4m}}{2}, \tag{114}$$

and they are complex, so that the mode oscillates, when $\gamma^2 < 4m$. For distinct roots the solution is

$$\xi_t = k_+ s_+^t + k_- s_-^t, \tag{115}$$

$$k_+ = \frac{\xi_1 - s_-\xi_0}{s_+ - s_-}, \qquad k_- = \frac{s_+\xi_0 - \xi_1}{s_+ - s_-}. \tag{116}$$

At a repeated nonzero root s, the limiting solution instead takes the form

$$\xi_t = (k_0 + k_1 t)s^t, \qquad k_0 = \xi_0, \quad k_1 = \xi_1/s - \xi_0. \tag{117}$$

The recurrence itself covers zero-root cases without division by a root.

For $0 \le m < 1$, the real second-order polynomial $s^2 - \gamma s + m$ has both roots inside the unit disk precisely when

$$1 - \gamma + m = \eta_h \mu > 0, \tag{118}$$

$$1 + \gamma + m = 2(1 + m) - \eta_h \mu > 0, \tag{119}$$

$$1 - m > 0. \tag{120}$$

Thus the stability interval is $0 < \eta_h \mu < 2(1 + m)$. These formulas specify the heavy-ball solver. Nesterov implementations use their own gradient-evaluation point and corresponding recurrence; the experimental protocol identifies the solver used for each network configuration.

E.1.10 Complete-gradient response identities

Substituting $d_l = \epsilon_l + q_l$ into the two local rules gives the general identity of Eq. (42): the first term carries the changed prediction and the second the changed parameter Jacobian. Proposition 3 gives the corresponding target differential for the affine readout.

The directional expansion of the finite-step residual operator uses the left singular vectors ϕ_j of M_z. Writing a unit direction as $v = \sum_j \langle \phi_j, v\rangle \phi_j + v_\perp$, with $v_\perp \perp \mathrm{col}(M_z)$, gives

$$\pi_T(v)^2 = \|(I - P_T)v\|^2 = \sum_j (1 - \eta_h \mu_j)^{2T} \langle \phi_j, v\rangle^2 + \|v_\perp\|^2. \tag{121}$$

E.2 Interpolation geometry and local control

The intermediate rule mixes saved inputs at every learned layer as in Eq. (7). In the paper's convention, $\lambda = 0$ reads the inferred input (PC) and $\lambda = 1$ reads the forward input (PC+F), as in the experiment records.

For the affine readout, hold h_{L-1}^T and h_{L-1}^0 fixed during weight differentiation. At the input $h_{L-1}^\lambda = (1 - \lambda)h_{L-1}^T + \lambda h_{L-1}^0$ the local residual is

$$\epsilon_L^\lambda = y - V h_{L-1}^\lambda - b = (1 - \lambda)\epsilon_L + \lambda d_L, \tag{122}$$

$$g_V^\lambda = \nabla_V \tfrac{1}{2}\|\epsilon_L^\lambda\|^2 = -\epsilon_L^\lambda (h_{L-1}^\lambda)^\top \tag{123}$$

$$= -(1 - \lambda)^2 \epsilon_L (h_{L-1}^T)^\top - \lambda^2 d_L (h_{L-1}^0)^\top - \lambda(1 - \lambda)\big[\epsilon_L (h_{L-1}^0)^\top + d_L (h_{L-1}^T)^\top\big]. \tag{124}$$

The bias gradient is $-\epsilon_L^\lambda$. Mixing the endpoint gradients instead would give $(1 - \lambda)g_V^{\mathrm{PC}} + \lambda g_V^{\mathrm{F}}$. The difference is

$$g_V^\lambda - \big[(1 - \lambda)g_V^{\mathrm{PC}} + \lambda g_V^{\mathrm{F}}\big] = \lambda(1 - \lambda)(\epsilon_L - d_L)\, d_{L-1}^\top. \tag{125}$$

The two expressions agree at PC and PC+F, while the interior contains the cross term because both the residual and the local parameter Jacobian change. The experiment uses g_V^λ, not a mixture of endpoint gradients.

The size of the mixed residual,

$$\|\epsilon_L^\lambda\|^2 = (1 - \lambda)^2 \|\epsilon_L\|^2 + \lambda^2 \|d_L\|^2 + 2\lambda(1 - \lambda)\langle \epsilon_L, d_L\rangle, \tag{126}$$

depends on the angle between the PC and PC+F endpoint signals as well as on their lengths.

E.3 NUMERICAL CHECKS

The analytic results were checked against independent numerical calculations in a quadratic model with a 10×64 matrix whose singular values range from 0.5 to 3. Analytic states, target sensitivities and complete weight gradients were compared with independent numerical calculations; the finite-difference error of the weight gradient is of order 10^{-10}. In the trained networks, the exact identities used in the probes are checked on saved vectors: the label-change gradient decomposition, the affine hidden-clamp prediction and the basis reconstructions (Tables 13 and 15).

F APPLIED UPDATES AND LAYER-SCALED BP

This appendix follows the raw gradient differences of Section 5.2 into the updates applied by the optimizer and compares PC with a layer-scaled BP baseline.

F.1 SMALL LOCAL ERRORS AND APPLIED UPDATES

Inference leaves PC with much smaller local errors than PC+F, and hence much smaller raw gradients (Section 5.2). This raises a direct objection to the behavioral result: PC might retain few wrong labels simply because it barely changes its weights. The experiment below asks whether a small raw gradient still yields a small applied update under the optimizer used in training.

Each branch starts from the same PC checkpoint (epoch 25, seed 0) with the inherited AdamW state and continues for 50 updates on the same sequence of corrupted-example batches, recomputing inference at its own current weights. SGD branches without momentum serve as the reference in which the applied update is proportional to the raw gradient. Table 17 reports the raw gradient norm, the applied update norm, and the change in the network's predictions.

Optimizer	Rule	$\|g\|$, step 1	$\|u\|$, step 1	c @50	Test @50
SGD	PC	0.0204	0.0026	0.632	0.698
SGD	PC+F	0.2966	0.0376	0.503	0.598
AdamW	PC	0.0204	0.0298	0.226	0.342
AdamW	PC+F	0.2966	0.0376	0.209	0.319

Table 17: **Small local errors do not imply small applied updates.** Branches continue the same PC checkpoint (epoch 25, seed 0) for 50 updates on corrupted examples. $\|g\|$ is the raw gradient norm and $\|u\|$ the norm of the update the optimizer applies. Both norms come from a one-step probe on a shared first batch. The SGD step size is set so that the first PC+F update matches its AdamW counterpart. At the start, $c = 0.634$ and test accuracy is 0.699. The raw gradients differ by $14.5\times$; after AdamW the first applied updates differ by $1.26\times$.

Under SGD the small PC gradient translates into a small update, and the PC branch leaves its predictions essentially unchanged over the 50 updates. Under AdamW the same gradient produces an applied update of the same order as that of PC+F, and after 50 updates the PC branch has changed its predictions as much as the PC+F branch. With the optimizer used in training, a small local error therefore does not prevent large weight changes.

Two further results point the same way. PC fits most of a large pool of fixed random labels under the same optimizer and update budget (Section 3.2), and under asymmetric corruption it memorizes most wrong labels while its label-replacement gradients remain attenuated (Appendix G.2). The restrained memorization under symmetric noise is therefore not a consequence of small updates. It has to come from properties that the optimizer's normalization preserves: the direction of the label-induced response (Figure 9b) and the way per-example responses combine across examples (Appendix G.1.1).

A corrupted-only stream is suited to this question but not to measuring memorization. Every label in it differs from the true class, so its averaged gradient mainly suppresses the true class: under AdamW, recovery c and test accuracy fall while predictions move to classes that are neither true nor noisy, and most evaluated images are not revisited within the 50 updates. We therefore report prediction change rather than wrong-label fitting.

F.2 LAYER-SCALED BP

The BP frontier applies layerwise scales to AdamW updates and retunes global weight learning rate and decay. Its selected profile reduces updates in earlier layers while preserving the readout scale. The resulting fitting and test metrics are recorded in Table 18.

Corruption	Final m_c	Final m_n	Test (R0)
0.4	0.9333	0.2037	0.7597
0.6	0.8710	0.2603	0.6283

Table 18: Five-seed means for the selected BP frontier profile $[0.05, 0.05, 0.3, 0.8, 1.0]$. Learning rate 9.6983×10^{-4}; weight decay 5.2455×10^{-3}; 50 training epochs. Selection retunes a separate BP protocol using the campaign's noisy held-out objective.

The selected profile keeps strong clean fitting while restraining noisy-label fitting, and at corruption 0.4 its endpoint is comparable to that of PC in the central benchmark. Restraint of noise memorization is therefore not exclusive to PC. The selection used a noisy held-out objective, so the profile was chosen for this restraint, and it reaches it through the opposite layer profile: it shrinks updates to early layers and keeps the readout, whereas in PC the attenuation of the label-induced response relative to PC+F is confined to the readout (Figure 9c).

G NOISE TYPES AND CORRUPTION RATES

G.1 NOISE STRUCTURE AND CROSS-EXAMPLE AGGREGATION

Repeated updates depend on which incorrect associations recur across images. Under the conditions in Figure 17, PC memorizes more class-structured asymmetric labels than symmetric labels, and its endpoint gap from BP narrows. A class-dependent wrong mapping repeats across examples; symmetric flips distribute the errors more broadly. In the accompanying asymmetric probes, substantial memorization coexists with attenuated per-image label-replacement gradients.

G.1.1 AGGREGATION AND COHERENCE

To connect per-image responses to a batch update, let g_i denote a per-example gradient at fixed weights. Its average obeys

$$\left\| \frac{1}{N} \sum_i g_i \right\|^2 = \frac{1}{N^2} \sum_i \|g_i\|^2 + \frac{1}{N^2} \sum_{i \neq j} \langle g_i, g_j \rangle. \tag{127}$$

The first term records individual magnitudes; the second records reinforcement and cancellation across examples. A normalized form of the same decomposition is the coherence

$$\kappa = \frac{\| \sum_i g_i \|^2}{N \sum_i \|g_i\|^2}, \qquad \sum_i \|g_i\|^2 > 0. \tag{128}$$

Writing $\bar{g} = N^{-1} \sum_i g_i$ and $g_{\mathrm{rms}}^2 = N^{-1} \sum_i \|g_i\|^2$ gives

$$\|\bar{g}\|^2 = \kappa g_{\mathrm{rms}}^2, \tag{129}$$

$$\kappa = \frac{1}{N} + \frac{\sum_{i \neq j} \langle g_i, g_j \rangle}{N \sum_i \|g_i\|^2}. \tag{130}$$

Cauchy–Schwarz gives $0 \leq \kappa \leq 1$. The upper bound is attained when all vectors are identical, while cancellation can make the numerator vanish. Thus the statistic describes how individual magnitudes combine into a shared update, including both alignment and norm heterogeneity.

Under true targets, the per-example readout gradients of PC largely cancel within a class, whereas those of PC+F add up; label-replacement responses are also less coherent under PC. Because κ does not depend on the overall scale of the per-example gradients, this difference is not a consequence

Class-conditioned output vector	PC	PC+F
Label replacement $g_i^n - g_i^c$	0.2064	0.2922
True-target gradient g_i^c	0.0495	0.3702

Table 19: Mean class-conditioned coherence of readout gradients at the PC epoch-25 checkpoint of the shared configuration. Both rows use the same corrupted-image pool; the true-target row changes its labels for inference.

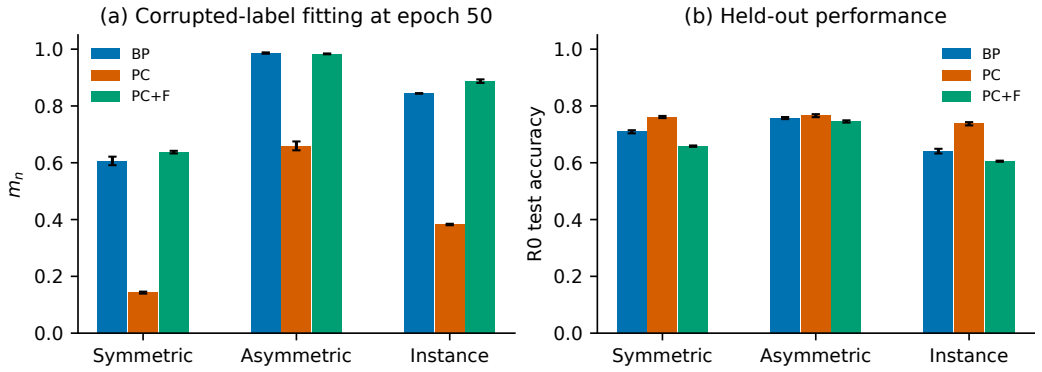


Figure 17: **Wrong-label fitting and test accuracy across noise types.** (a) Final wrong-label fitting; (b) mean test accuracy over the last ten evaluations.

of the small PC errors, and an overall rescaling of the update, which AdamW largely provides here (Appendix F.1), leaves it in place. The measurement uses one checkpoint and one training seed.

The per-example transfer measurements are the same kind of evidence under both noise types, but they predict memory only together with the structure of the errors; Appendix G.2 gives the paired measurements.

G.2 Continuation and corruption-structure follow-ups

G.2.1 Stable continuations on a reference data stream

The stable SGD follow-up starts both rules from each shared checkpoint and uses a common calibrated learning rate within each continuation condition. Its mixed stream samples reference clean and corrupted examples equally. Calibration selects a stable rate using a short continuation and a clean-fit tolerance. Table 20 reports the subsequent endpoint for the original two rules and a forward-input branch whose total update norm matches PC at each step.

Checkpoint family	Seed	PC	PC+F	PC+F, total-norm matched
PC	0	0.1489 / 0.8677	0.1519 / 0.8477	0.1475 / 0.8682
PC	1	0.0278 / 0.7412	0.0420 / 0.6885	0.0293 / 0.7393
PC	2	0.0298 / 0.7456	0.0347 / 0.7393	0.0283 / 0.7451
PC+F	0	0.3628 / 0.8735	0.2729 / 0.9038	0.2705 / 0.9097
PC+F	1	0.0610 / 0.6606	0.0386 / 0.7646	0.0366 / 0.7856
PC+F	2	0.0605 / 0.6880	0.0366 / 0.7617	0.0352 / 0.7837

Table 20: Noisy fit / clean fit after 500 mixed-stream SGD updates from epoch-25 checkpoints. Metrics refer to the fixed reference evaluation pools. Total-norm matching is checked from actual recorded update norms.

On PC-trained checkpoints, total-norm matching brings the continuation endpoints close to the PC branch. On PC+F-trained checkpoints, switching to the inferred-input rule raises noisy fitting. The filtering effect depends on the learned weights as well as the input read by the local rule.

G.2.2 Asymmetric memorization alongside gradient attenuation

The new shared-configuration asymmetric runs pair training trajectories with target-conditioned probes at their saved checkpoints. PC learns the structured wrong labels while its label-replacement gradients remain attenuated relative to PC+F at the same inferred states (Table 21). Strong per-example attenuation therefore coexists with substantial accumulated memorization.

Seed	Asym. rate	Sym. rate	Asym. m_n	Sym. m_n	Asym. $\bar{\rho}_{\mathrm{all}}$
0	0.1578	0.3967	0.8135	0.2043	0.0543
1	0.1624	0.4005	0.8136	0.1358	0.0531
2	0.1617	0.4018	0.8005	0.1443	0.0510

Table 21: PC epoch-50 training endpoints and mean per-image whole-model label-replacement norm ratios. Rates are realized fractions of changed training labels. Asymmetric and symmetric runs share nominal corruption settings and training hyperparameters. Asymmetric probes use 109, 42, and 44 corrupted images for seeds 0, 1, and 2, respectively.

These trajectories distinguish an instantaneous response from its accumulation over training. Relating accumulation to corruption structure requires the cross-example sums and alignments defined in the aggregation analysis.

G.3 Human-annotation results

The annotation comparison evaluates the central methods on CIFAR-10N (Wei et al., 2022), whose labels come from human annotators and whose errors depend on the image and the annotator. Table 22 reports terminal fitting together with both terminal and validation-selected test performance.

Annotation	Method	Seeds	m_c (50)	m_n (50)	Test (R0)	Test (ES)
agg	PC	3	0.961 ± 0.001	0.344 ± 0.007	0.845 ± 0.002	0.844 ± 0.002
agg	PC+F	3	0.997 ± 0.003	0.820 ± 0.080	0.841 ± 0.004	0.841 ± 0.005
agg	BP	3	0.999 ± 0.000	0.844 ± 0.003	0.864 ± 0.002	0.863 ± 0.005
random1	PC	3	0.953 ± 0.001	0.265 ± 0.003	0.828 ± 0.003	0.829 ± 0.004
random1	PC+F	3	0.997 ± 0.000	0.783 ± 0.007	0.800 ± 0.003	0.803 ± 0.005
random1	BP	3	0.998 ± 0.000	0.763 ± 0.004	0.829 ± 0.001	0.839 ± 0.004
random2	PC	3	0.954 ± 0.001	0.253 ± 0.004	0.827 ± 0.001	0.826 ± 0.003
random2	PC+F	3	0.997 ± 0.000	0.778 ± 0.004	0.798 ± 0.003	0.801 ± 0.003
random2	BP	3	0.998 ± 0.000	0.748 ± 0.005	0.825 ± 0.000	0.833 ± 0.004
random3	PC	3	0.954 ± 0.002	0.265 ± 0.007	0.827 ± 0.002	0.826 ± 0.004
random3	PC+F	3	0.799 ± 0.342	0.579 ± 0.358	0.652 ± 0.255	0.803 ± 0.005
random3	BP	3	0.997 ± 0.000	0.761 ± 0.002	0.825 ± 0.001	0.833 ± 0.003
worst	PC	3	0.936 ± 0.000	0.290 ± 0.002	0.755 ± 0.001	0.753 ± 0.006
worst	PC+F	3	0.987 ± 0.001	0.742 ± 0.003	0.658 ± 0.005	0.726 ± 0.003
worst	BP	3	0.990 ± 0.001	0.721 ± 0.003	0.696 ± 0.002	0.763 ± 0.004

Table 22: **Completed human-annotation experiments.** GELU VGG5/CIFAR-10N cells from the same benchmark campaign; means and sample SD over the stated seeds. Random annotations are kept separate. A terminal decline remains in the reported average rather than being removed as an outlier.

Human noise reproduces the pattern of symmetric noise. In every annotation condition PC ends training with far lower wrong-label fitting than BP and PC+F. Its terminal test accuracy matches or exceeds that of BP in the random-annotator conditions and exceeds it clearly in the worst-annotator condition, while at the clean-validation-selected checkpoint BP is higher in every condition. As under symmetric noise, the advantage of PC lies in what continued training adds rather than in the best checkpoint along the path. One PC+F run in the random3 condition deteriorates late in training and produces the large terminal spread of that cell; it is kept in the average.

H PC VARIANTS AND OPTIMIZATION SETTINGS

H.1 BENCHMARK TRAJECTORIES OF PC VARIANTS

The inference rule, applied updates, and data stream jointly shape memorization. ePC uses an error-based reparameterization (Goemaere et al., 2025); PC+SF is PC+F with the spiking precision of Qi et al. (2026), which raises the precision of each layer at the inference step when the error signal first reaches it, so that the energy propagates to lower layers within the first L steps. Under their benchmark configurations, Figure 18 shows that both establish clean fitting and then continue to acquire wrong labels, accompanied by a test-accuracy decline.

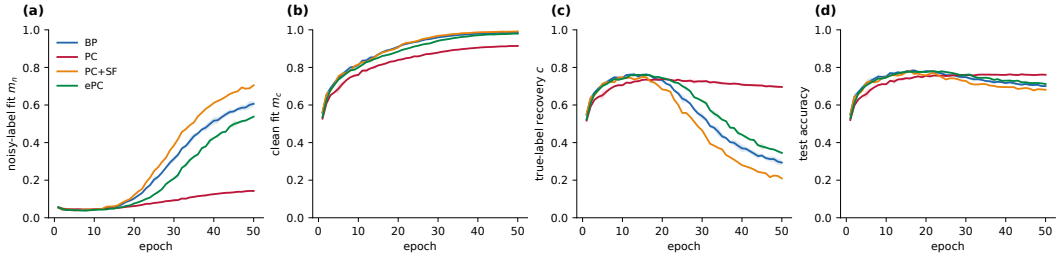


Figure 18: **Training trajectories of PC variants.** Panels show wrong-label fitting, clean fitting, recovery, and test accuracy. BP, PC, and ePC summarize three seeds; PC+SF is one archived run.

Figure 19 and Table 23 show how matched weight-rate and batch-size comparisons further change the ePC trajectory.

H.2 EPC, PC+SF, AND OPTIMIZATION SETTINGS

The variant comparison asks how training trajectories differ when the models reach similar clean-validation-selected performance. All runs use VGG5/CIFAR-10, symmetric corruption 0.4, and three seeds. BP and PC+SF share the selected weight learning rate and batch size but retain their own weight decay. The matched ePC arm additionally uses AdamW and BP's decay. Table 23 includes other ePC optimization and error-learning-rate settings.

Method (optimizer)	η_w	Batch	test@ES	ES epoch	m_n final	c final	test last10
BP (AdamW)	$2.788\,10^{-4}$	128	0.7860 ± 0.0025	16–22	0.606	0.294	0.709
PC+SF (AdamW)	$2.788\,10^{-4}$	128	0.7840 ± 0.0030	13–21	0.603	0.298	0.712
ePC (Adam+L2, $e_{\mathrm{lr}}{=}10^{-3}$)	$4.708\,10^{-4}$	256	0.7828 ± 0.0047	21–26	0.538	0.345	0.721
ePC (AdamW, $e_{\mathrm{lr}}{=}10^{-3}$)	$2.788\,10^{-4}$	128	0.7846 ± 0.0083	16–20	0.501	0.381	0.728
ePC (AdamW, $e_{\mathrm{lr}}{=}10^{-4}$)	$2.788\,10^{-4}$	128	0.7872 ± 0.0010	16–22	0.577	0.319	0.722

Table 23: ePC optimizer, learning-rate, and batch-size comparisons. BP, PC+SF, and the last ePC row share AdamW, the weight learning rate, and batch size; ePC also uses BP's weight decay. The last row uses error learning rate 10^{-4}. Each row summarizes three seeds. Test@ES is mean and sample SD; other metrics are means. Figure 19 plots BP, PC+SF, the last ePC row, and baseline PC.

BP, PC+SF, and matched ePC reach similar test accuracy at the selected checkpoint, but finish with different wrong-label fitting and recovery. Every ePC arm ends below BP and PC+SF in wrong-label fitting. Among the ePC arms the error step size matters more than the optimizer: at $e_{\mathrm{lr}} = 10^{-3}$ the Adam+L2 arm with batch 256 and the AdamW arm with batch 128 finish close together, while reducing e_{lr} to 10^{-4} moves the AdamW arm toward BP. These comparisons jointly vary learning rate, batch size, and decay semantics, as recorded in the table.

The error-learning-rate sweep further separates optimization regimes. Reducing a positive error learning rate brings some noisy-label trajectories closer to BP; on clean data, intermediate rates form a performance plateau. Very small rates lose hidden-update resolution in floating-point arithmetic, while rates beyond the stability region diverge. The implementation's normalization is singular at literal zero, so these comparisons use positive rates.

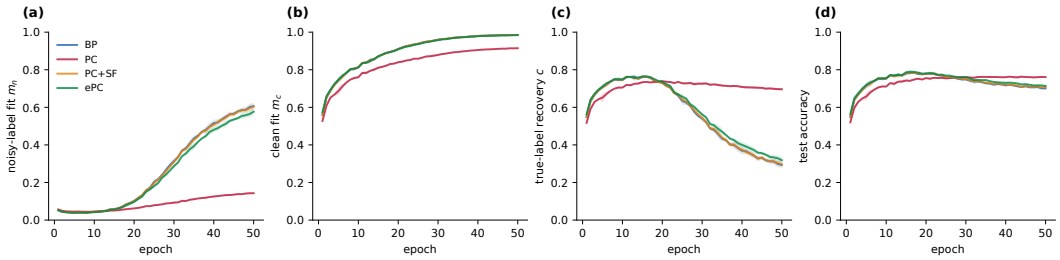


Figure 19: **Variant trajectories under shared weight-learning rate and batch size.** BP, PC+SF, and ePC use the stated common settings; baseline PC is shown for reference. Endpoint metrics and remaining settings appear in Table 23.

H.2.1 Error-inference step-size sweep

We vary the error-inference step size with the number of inference iterations fixed. Within each series, the weight optimizer, learning rate, decay, and batch size remain fixed. The first series uses AdamW with the BP reference settings; the second uses Adam with coupled L2 decay and its recorded settings. The execution scripts install these optimizers; the AdamW raw export retains an inherited optimizer-name field. In the ePC implementation, inferred errors are added to each hidden-layer prediction to reconstruct the state read by the local weight rule. The local weight objective is divided by $\min(1, Te_{\mathrm{lr}})$, so changing e_{lr} changes both the inferred state and this gradient normalization.

The small-step relation in Eq. (8) follows directly from the error update. Let $\mathcal{L}(h_L, y)$ be the output loss and initialize every hidden error at zero. At zero error, the gradient of the output loss with respect to e_l is the backward sensitivity $\delta_l^0 = \partial\mathcal{L}/\partial h_l^0$. For fixed T and differentiable predictions, the error descent recurrence gives

$$e_l^T = -Te_{\mathrm{lr}}\delta_l^0 + O(e_{\mathrm{lr}}^2), \qquad h_l^T = h_l^0 + O(e_{\mathrm{lr}}). \tag{131}$$

Consequently the normalized hidden-layer local gradient approaches $(J_{W,l}^0)^\top \delta_l^0$ as e_{lr} tends to zero through positive values. At finite step size, both $e_l^T/(Te_{\mathrm{lr}})$ and the inferred input in Eq. (8) change.

Optimizer	e_{lr}	Successful	Final m_n	Final m_c	Test last10
AdamW	0.0001	3/3	0.577 ± 0.019	0.984 ± 0.001	0.722 ± 0.001
AdamW	0.0005	3/3	0.549 ± 0.022	0.982 ± 0.001	0.724 ± 0.002
AdamW	0.001	3/3	0.501 ± 0.024	0.977 ± 0.001	0.728 ± 0.005
AdamW	0.005	2/3	0.060 ± 0.011	0.834 ± 0.054	0.760 ± 0.022
AdamW	0.01	0/3	–	–	–
Adam+L2	0.0001	3/3	0.595 ± 0.003	0.984 ± 0.000	0.716 ± 0.008
Adam+L2	0.0005	3/3	0.580 ± 0.007	0.982 ± 0.002	0.714 ± 0.004
Adam+L2	0.001	3/3	0.538 ± 0.004	0.980 ± 0.001	0.721 ± 0.003
Adam+L2	0.005	1/3	0.058	0.866	0.776
Adam+L2	0.01	0/3	–	–	–

Table 24: Complete ePC error-inference sweep. Metrics summarize successful runs, defined by mean test accuracy over the last ten epochs above 0.3. Successful counts include all three attempts. Entries give means and sample SD for multiple successes, a single value for one success, and – for none.

Increasing the error-inference step lowers wrong-label fitting in both series. Clean fitting remains high throughout the interval where all runs succeed. At the next step size, successful runs show a further reduction in noisy fitting together with lower clean fitting and higher test accuracy. Figure 7 and Table 24 summarize these successful runs; the table also reports their frequency among all attempts. At the largest step size no run meets the success criterion. The sweep therefore separates the fitting behavior of trained networks from the stability of training as inference steps grow. Its small-step end agrees with Eq. (131): as e_{lr} decreases, the normalized ePC update tends to the BP gradient at forward states, and the noisy-label endpoint approaches that of BP. Changing the optimizer semantics, learning rate and batch size at fixed e_{lr} shifts the endpoint less than the step-size sweep does (Tables 23 and 24), so the step size is the main control of ePC noise memorization in this range.

I RELATED WORK

Predictive coding as inference followed by learning. Predictive coding began as an account of cortical prediction and free-energy minimization (Rao & Ballard, 1999; Friston, 2010; Bogacz, 2017) and has since been developed into a local learning algorithm for deep networks (Millidge et al., 2022a; Salvatori et al., 2025). The rule studied here first lets the states settle under the clamped input and target and then updates each weight to reduce a local prediction error at the settled states; incremental variants instead interleave weight updates with inference (Salvatori et al., 2024). Benchmark studies describe how this rule trains at scale (Pinchetti et al., 2025), and recent work addresses its degradation with depth through precision schedules and a modified weight update (Qi et al., 2026), depth-aware parameterization (Innocenti et al., 2026b), error-based inference (Goemaere et al., 2025), and initialization (Pinchetti et al., 2026). We use two of these rules, Forward Update and ePC, as interventions on the weight update.

When the local update reproduces backpropagation. PC approximates BP when the output error is weighted weakly relative to the feedforward prediction (Whittington & Bogacz, 2017), along arbitrary computation graphs under a fixed-prediction assumption (Millidge et al., 2020), and exactly when weights are updated at prescribed inference steps (Song et al., 2020; Salvatori et al., 2021b; Rosenbaum, 2022). Millidge et al. (2022b) unify these results with equilibrium propagation and contrastive Hebbian learning as the infinitesimal-inference limit of energy-based models, and Innocenti et al. (2026a) show that, at activity equilibrium and under stable, feature-learning parameterizations, the PC gradient converges to the BP gradient when the network is much wider than deep. In these results the local update approaches BP because the states it reads stay close to the feedforward pass, or because errors are read at controlled steps. Our analysis holds inference fixed and varies the state that the weight update reads.

Learning away from these limits. Song et al. (2024) describe standard PC learning as prospective configuration, in which activities first move toward the values that would produce the target and the weights then consolidate them. This regime has been related to generalized expectation maximization and to an interpolation between feedforward activities and target-propagation targets (Millidge et al., 2022c), to implicit SGD (Alonso et al., 2022), to second-order information (Alonso et al., 2024), and to trust-region steps (Innocenti et al., 2023). Two results are closest to ours.

The first is a comparison of weight inputs by Alonso et al. (2022). In linear networks trained one example at a time with closed-form Gauss–Newton targets, they compare an update that reads the inferred presynaptic state with a Gauss–Newton backpropagation update that reads the feedforward presynaptic activity. The latter has the form of PC+F, with local targets obtained from Gauss–Newton target propagation rather than from PC inference. Under sign conditions on the inferred and feedforward activities, the inferred-state update moves the feedforward output along the negative loss gradient for any positive learning rate, whereas the feedforward-state update does so only for a bounded range of learning rates (their Theorems D.1 and D.2). Their question is the stability of a single update; ours is which target information each reading passes into the weights of trained nonlinear networks, and what the networks consequently memorize from incorrect labels.

The second is the closed form of the output residual after inference in deep linear networks. The equilibrated energy is the rescaled mean squared error $\frac{1}{2} d_L^\top \Gamma^{-1} d_L$ with $\Gamma = I + \sum_\ell W_{L:\ell} W_{L:\ell}^\top$ and $d_L = y - h_L^0$ (Innocenti et al., 2024), so that $\epsilon_L = \Gamma^{-1} d_L$ at equilibrium, and Oliviers et al. (2026) show that this factor removes the weight-dependent distortion that BP applies to the output residual. The equilibrium expression of Proposition 4 corresponds to the single-hidden-layer case of Γ^{-1}. Proposition 2 concerns a different quantity: the response after the finite number of inference steps used in training, which reaches the equilibrium expression only as the number of steps grows (Eq. (103)). The analysis also keeps the residual and the input at which the parameter Jacobian is evaluated as separate factors.

Other rules that learn from a controlled state. Equilibrium propagation contrasts a free and a weakly nudged state and recovers the gradient as the nudging vanishes (Scellier & Bengio, 2017); the least-control principle treats a hard-clamped output as perfect control and shows that the PC update then descends a least-control objective (Meulemans et al., 2022); bilevel formulations relate these rules to implicit differentiation (Zucchet & Sacramento, 2022). Target propagation trains each

layer to move its forward activity toward a local target produced by learned inverses (Bengio, 2014; Lee et al., 2015; Meulemans et al., 2020). PC+F has the form of a target-propagation update whose targets come from PC inference: its local loss compares an inferred target with the prediction from the forward input. These rules differ in which state the weight update reads, and the mixed-input rule of Eq. (7) moves continuously along this choice.

Memorization and label noise. Networks trained with BP can fit random labels (Zhang et al., 2017), learn shared structure before individual exceptions (Arpit et al., 2017; Toneva et al., 2019), and memorize atypical examples as part of generalization (Feldman, 2020; Carlini et al., 2019). Two mechanistic accounts are closest to ours. Li et al. (2020b) show that with a low-rank Jacobian the clean residual lies in the high-gain subspace and label corruption in the low-gain subspace, so gradient descent fits the corruption only late. Stephenson et al. (2021) locate memorization of permuted labels in the final layers and trace its delay to the small gradient contribution of permuted examples early in training, while Maini et al. (2023) find memorized examples distributed over a few neurons across layers, with clean and noisy gradients strongly opposed. In these accounts the filtering acts across training time through the parameter Jacobian. We ask whether inference in PC adds a filter that acts within each example, before any weight change. Practical remedies select samples, correct the loss, or regularize toward early predictions (Song et al., 2022; Natarajan et al., 2013; Patrini et al., 2017; Han et al., 2018; Li et al., 2020a; Liu et al., 2020), and CIFAR-10N supplies human annotation errors for evaluation (Wei et al., 2022). We use no noise-specific method; the restraint we study arises from the learning rule. Predictive coding networks also store patterns as associative memories in their weights (Salvatori et al., 2021a); label noise provides a supervised setting in which the association to be stored is known to be wrong.

J Discussion

Where the filtering happens. The paired true/wrong probes locate the filtering step at the readout. For the same image, the wrong label creates a larger mismatch with the forward prediction, yet hidden-state adaptation absorbs it and leaves a readout residual of the same size as under the true label. The change in the residual caused by replacing the label is therefore small, and reading the inferred input changes the magnitude and direction of the label-induced gradient further. The direction probes give the geometry of this step: label replacements lie in the centered class-contrast subspace, which inference attenuates strongly, whereas a common output shift survives. In the equilibrium theory of deep linear PC networks the output residual after inference is $\Gamma^{-1} d_L$ with a weight-dependent Γ (Innocenti et al., 2024; Oliviers et al., 2026); the measured response map is the finite-budget, nonlinear counterpart of that filter, with one high-gain mode and weak class-contrast modes.

The filter is built by training. Absorption is not a property of the update formula alone. Under identical inference settings, PC-trained checkpoints absorb nearly all of the forward mismatch while PC+F-trained checkpoints absorb almost none, and at PC checkpoints the contrast grows over training (Table 8). The finite-step operator explains why the same solver can behave so differently: at a fixed budget the fraction retained along mode j is $(1 - \eta_h \mu_j)^T$, so absorption depends on the curvature μ_j that the learned weights give the state energy. Reading inferred inputs and the weights that this reading produces reinforce each other, and at PC+F-trained weights the inferred-input rule loses most of its filtering (Table 14). The reduced noise memorization of PC is a property of this co-adapted pair.

Small errors, normal updates. Inference leaves PC with small local errors and therefore small raw gradients, which raises the possibility that PC retains few wrong labels only because it barely updates its weights. The optimizer comparison rules this out for the optimizer used in training. Under SGD the small PC gradient gives a correspondingly small update and the predictions do not move; under AdamW the applied update is of the same order as that of PC+F, and the network's predictions change as much (Table 17). PC also fits large pools of random labels (Section 3.2) and memorizes most wrong labels under asymmetric corruption despite attenuated label-replacement gradients (Appendix G.2). The restraint under symmetric noise must therefore come from what the optimizer's normalization preserves: the direction of the label-induced response and the way per-example responses combine across examples. The difference between complete training runs also involves the weights themselves, since PC and PC+F build checkpoints that absorb target mismatch to very different degrees (Table 8). Structured corruption shows the limit of this restraint: when the

same wrong mapping recurs across images, attenuated responses still accumulate into substantial memory (Appendix G.1).

Implications for learning rules. Reading inferred states in the weight update is what lets PC absorb incorrect targets, and this ingredient need not remain tied to PC. Predictive coding that reads inferred states throughout is still hard to scale: μPC trains networks with more than a hundred layers, but relies on a Depth-μP parameterization of fully connected residual networks and has been demonstrated on simple classification tasks (Innocenti et al., 2026b). Our interventions indicate that full PC is not required for the effect. In the mixed-input rule, reading half of the inferred input keeps test accuracy close to PC while fitting far fewer wrong labels than PC+F (Figure 6); in ePC, a larger error-inference step, with an update that otherwise stays close to BP, sharply lowers wrong-label fitting, although training becomes unstable at the largest successful step size (Table 24). The amount of state inference that enters the update is therefore a tunable trade-off between BP-like updates and the noise restraint of PC. Locating this trade-off, and adding a short target-conditioned state adjustment before the weight update of BP-trained networks, are concrete directions for new methods of learning from noisy labels.

K LIMITATIONS

Theory. The finite-step operator describes fixed local matrices under gradient-descent inference, while the trained networks have nonlinear state dynamics and a momentum solver. The coordinate probes reconstruct an empirical output response map at each checkpoint; they do not estimate the local Jacobians M_z and M_w, so the operator accounts for the direction probes qualitatively rather than predicting them from measured curvature. Extending inference beyond the training budget changes the readout residual non-monotonically (Table 16), so the account holds for the budget used in training and not for the equilibrium limit.

What the probes measure. All transfer measurements compare gradients in parameter space. They do not measure the resulting change in the network's predictions, which is the quantity that target-alignment analyses use (Oliviers et al., 2026). Measuring the one-step change in the forward output under true and wrong labels would connect the two descriptions directly.

Scale and performance. Because PC itself degrades in deep models, the experiments use VGG5 and MLP5 on CIFAR-10, CIFAR-100 and Fashion-MNIST under the standard parameterization. On CIFAR-100 PC loses clean fitting under heavy corruption, so the behavioral contrast is established only where PC trains stably. At clean-validation-selected checkpoints BP reaches higher test accuracy than PC at every symmetric corruption rate and in every CIFAR-10N condition, and under symmetric noise the two methods fit similar fractions of wrong labels there. The advantage of PC lies in what continued training adds, not in the best checkpoint.