

Comparing the Sample and Computational Efficiency of Unit-Precision Predictive Coding and Backpropagation

Austin Huang

Eton College

Windsor, Berkshire, United Kingdom

austinhuang1121@outlook.com

Abstract

Synchronous unit-precision predictive coding (PC) replaces backpropagation's (BP's) immediate backward pass with recurrent activity inference. We compared PC with objective- and optimizer-matched BP in tanh multilayer perceptrons on MNIST and Fashion-MNIST. In the depth-2 network trained on 100 labelled MNIST examples, each method's learning rate was selected independently using validation data, and both searches met the prespecified bracketing rule. With 200 inference steps, PC came close to BP's accuracy but remained lower. Across ten fresh paired seeds, the mean epoch-30 PC-minus-BP test-accuracy difference was -0.295 percentage points (95% paired-bootstrap interval $[-0.376, -0.181]$). PC nevertheless required substantially more computation. Diagnostic-free float32 training took 130.6 times as long as BP on a single CPU thread, while analytical counts gave approximately 113 times as many MACs, 112 times as many FLOPs, and 403 times as many tanh evaluations. The depth-2, 500-label Fashion-MNIST comparison remained unresolved because the BP learning-rate search did not show the required numerically valid decline at a higher rate; its result is therefore descriptive. These findings apply only to the tested synchronous unit-precision formulation, datasets, architecture, and digital CPU implementation, not to all predictive-coding variants or hardware.

Keywords

Predictive coding, backpropagation, sample efficiency, computational efficiency, iterative inference, implementation validation

ACM Reference Format:

Austin Huang. 2026. Comparing the Sample and Computational Efficiency of Unit-Precision Predictive Coding and Backpropagation. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 3, ISSUE 4)*. ACM, New York, NY, USA, 12 pages. <https://doi.org/10.67149/yhjs2024.5/3u76nbg5>

1 Introduction

Backpropagation is highly effective for training multilayer networks, but its globally coordinated error signals have motivated alternatives that may be more compatible with biological or local computation [1]. Predictive-coding networks instead refine neural activities through local prediction errors [11]. Under particular assumptions, their weight updates can approach BP gradients [17].

This paper is published under the Creative Commons Attribution-NonCommercial-NoDerivs 4.0 International (CC-BY-NC-ND 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.

IJSCAR VOL. 3, ISSUE 4

© 2026 The Author(s). Published by IJSCAR.

That mathematical link is important, but it leaves a practical question unanswered: does recurrent inference improve learning enough to justify its cost?

The answer depends on the algorithm being tested. This study concerns synchronous, unit-precision PC: every prediction error has unit precision, hidden beliefs are updated together from an unchanged state snapshot, and predictions are recomputed during inference. Precision-weighted, fixed-prediction, amortized, incremental, and error-coordinate methods can behave differently. We therefore use "PC" as shorthand for the tested formulation, not for the entire predictive-coding family.

An apparent efficiency difference can arise from an incorrect implementation, a different objective, unequal initialization, or incomplete hyperparameter selection. Sample and computational efficiency must also remain separate. Reusing a fixed labelled set creates more presentations without supplying more unique labels, while every PC inference step consumes work absent from BP.

We address these issues in four stages. First, the PC implementation is checked against automatic differentiation, finite differences, and an independent energy formulation. Second, raw MSE gradients are compared with matched BP gradients, separating directional alignment, magnitude, and convergence to the PC fixed point. Third, a broad benchmark evaluates depths, labelled-set sizes, and three inference budgets; that experiment is retained as descriptive evidence because its learning-rate grid was truncated before validation turnover. Fourth, a prospective learning-rate extension uses a wider fixed grid, deterministic infill, independent validation selection, 200 inference steps, and untouched test evaluation on ten fresh paired seeds.

For the depth-2, 100-label MNIST condition with PC at $T = 200$, the wider search found numerically valid declines in validation accuracy on both sides of each method's near-best rates. PC's mean accuracy remained below BP's. The measured gap was smaller than in the separate original experiment, but the searches, inference budgets, and seeds differed, so that comparison does not identify the cause. Fashion-MNIST remained unresolved because BP never showed the required valid decline at a higher learning rate. Diagnostic-free CPU timing and analytical operation counts also show substantial recurrent cost. These findings apply to this synchronous unit-precision implementation rather than PC as a whole.

2 Theoretical and empirical context

Whittington and Bogacz related supervised PC updates to BP under assumptions that include small output errors and converged neural activities [17]. The model evaluated here assigns unit precision to every prediction error and does not hold predictions fixed during

inference. It therefore lacks the output-precision hierarchy or fixed-prediction construction that can bring gradient magnitudes closer together. Later analyses distinguish gradient direction from numerical equality and strict PC from modified inference rules [12, 13]. Algorithms that compute exact reverse derivatives form a separate category [7, 14].

Finite inference matters because convergence depends on the weights, initialization, and step size; outside suitable regimes, activity dynamics may diverge [2]. Validation must therefore distinguish convergence to the PC fixed point from agreement with BP. Reaching the former does not imply equality with the latter.

Recent work explores settings in which PC may learn efficiently, including deep or narrow architectures, pretraining, and conditioning interventions [8]. Other methods alter the computation through specialized software, hybrid or incremental schedules, depth-scaled parameterizations, or error-coordinate dynamics [3, 4, 10, 15, 16]. In particular, the current version of ePC is an error-based reparameterization designed for digital simulation and is reported as accepted to the ICML 2026 main track [3]. These approaches may reduce inference cost or change optimization behaviour, but their benefits cannot be assumed for conventional synchronous state inference, and local parameter updates alone do not guarantee lower end-to-end cost [19].

We evaluate only synchronous unit-precision PC and do not claim that the observed deficit is specific to this formulation. A comparison with a precision-weighted or fixed-prediction variant could determine whether the result changes under those interventions, but that experiment was not performed here. Such variants therefore remain an explicit limitation and future direction.

3 Methods

3.1 Data, matching, and original benchmark

Experiments used the official MNIST [6] and Fashion-MNIST [18] splits. Images were flattened to 784 dimensions and normalized using statistics from the corresponding training split. Split seed 999 selected a fixed 5,000-example validation pool from the 60,000 training examples. Model selection used training and validation data only; official test outcomes were accessed only after each configuration was finalized.

Every model was a width-256 tanh multilayer perceptron with biases, a linear 10-unit output, batch size 64, float32 training, and 30 epochs. Matched conditions shared initial parameters, examples, minibatches, and epoch order. Both theoretical comparators used mean-per-example MSE and plain SGD without momentum or weight decay. A conventional BP classifier with cross-entropy and Adam was retained as a practical reference, not as a theoretical comparator [5]. Models were implemented in PyTorch [9].

The original benchmark crossed MNIST depths two, four, and six with 100, 500, 1,000, and 5,000 labelled examples; Fashion-MNIST used depths two and six with 500 and 5,000 examples. PC inference budgets were $T \in \{10, 20, 50\}$, and ten paired seeds were used. This produced 48 PC-budget comparisons. Its six-rate learning-rate search ended at 0.03 while validation performance was still improving. We retain those results as broad descriptive evidence rather than as globally tuned comparisons.

3.2 Synchronous unit-precision PC

Let v_0 denote the input and v_K the output. With PyTorch’s parameter orientation,

$$\mu_{b,l+1} = \phi_l(v_{b,l})W_l^\top + b_l, \quad \epsilon_{b,l+1} = v_{b,l+1} - \mu_{b,l+1}, \quad (1)$$

where b indexes examples, ϕ_0 is the identity, and hidden activations use tanh. The per-example energy is

$$E_b = \frac{1}{2} \sum_{l=0}^{K-1} \|\epsilon_{b,l+1}\|_2^2. \quad (2)$$

Input and target output are clamped. At inference step t , every hidden belief is updated synchronously from the same snapshot:

$$v_{b,l}^{(t+1)} = v_{b,l}^{(t)} - \eta_{\text{inf}} \left. \frac{\partial E_{\text{sum}}}{\partial v_{b,l}} \right|_{v^{(t)}}, \quad E_{\text{sum}} = \sum_b E_b. \quad (3)$$

After T steps, beliefs are detached and the parameter rule is the batch-mean derivative

$$\frac{\partial E_{\text{mean}}}{\partial W_l} = -\frac{1}{B} \sum_{b=1}^B \epsilon_{b,l+1}^\top \phi_l(v_{b,l}), \quad E_{\text{mean}} = B^{-1} \sum_b E_b. \quad (4)$$

Thus, activity inference is not accidentally divided by the batch size, whereas parameter learning has the same mean reduction as BP. The learning rule does not differentiate through the inference trajectory. Appendix A gives the activity derivative, bias rule, stationarity definition, and numerical conventions; Appendix B reports the numerical checks.

3.3 Prospective learning-rate extension

We widened the learning-rate search to check whether the best results were still at its boundary. We required numerically valid runs with lower validation accuracy at rates below and above the near-best group; a failed run did not count as evidence of a decline. This rule brackets the selected region within the tested rates but does not prove a global optimum.

The extension fixed four depth-2 cells before development: MNIST with 100 labels and Fashion-MNIST with 500 labels, each comparing BP–MSE–SGD with PC–MSE–SGD at $T = 200$ and $\eta_{\text{inf}} = 0.2$. Five new development seeds (31001–31005) and ten disjoint confirmatory seeds (41001–41010) preserved paired initialization, examples, and epoch order.

Every cell first evaluated the same 18 learning rates:

$$0.0003, 0.001, 0.003, 0.01, 0.03, 0.04, 0.05, 0.06, 0.07, \\ 0.08, 0.09, 0.1, 0.12, 0.15, 0.3, 1, 3, 10.$$

A candidate could enter selection only when all five development runs completed with finite values and passed the prespecified numerical checks. Selection maximized mean epoch-30 validation accuracy. Rates within 0.1 percentage points of the maximum formed the near-optimal set, and the smallest such rate was selected. Formally, a qualifying flank had to lie outside that set, fall at least 0.5 points below the maximum, and be worse than the selected rate in at least four of five paired seeds.

If a flank was absent after the fixed grid, up to three deterministic geometric-midpoint attempts per side filled the largest remaining log-rate gap. A cell was fully performance-bracketed only when

qualifying finite lower and upper flanks existed. A rate that succeeded in some seeds but failed in others had mixed feasibility; neither that pattern nor failure in all seeds counted as validation turnover. A cell still lacking the required evidence after the fixed actions was labelled unresolved. These labels were fixed before test access.

Both depth-2, 100-label MNIST method cells met the bracketing rule. On Fashion-MNIST, PC met the rule. BP had a learning rate that completed successfully in all five development runs, so it could be carried forward, but its search remained unresolved after three upper-side midpoint attempts. Both datasets entered confirmatory evaluation, but only the specified MNIST condition supports a bracketed comparison.

3.4 Analysis and resource accounting

The main quantity estimated in the extension was the paired epoch-30 test-accuracy difference, PC minus BP, within each dataset. We also measured performance across the full training run rather than only at epoch 30. For epochs 1–30, we calculated the area under each test-accuracy curve against cumulative training-example presentations, divided by the presentation range, and compared PC with BP within each paired seed. This is the normalized test-accuracy-versus-cumulative-presentation AUC difference used below. Ten-thousand paired seed-level bootstrap resamples produced percentile intervals; time ratios were bootstrapped on the log scale. These intervals are conditional on the selected rates, fixed split, and confirmatory design; they do not incorporate uncertainty from repeating the development search. No p -values or multiplicity-adjusted claims were used. Extension seeds were never pooled with the original benchmark or earlier construction attempts.

Timing used float32 PyTorch on one Apple ARM64 CPU thread. For each confirmatory run, an isolated disposable model warmed the cache; the untouched identically initialized model then supplied 30 diagnostic-free timed epochs. Warm-up, preprocessing, evaluation, diagnostics, serialization, and logging were excluded. These ratios characterize this implementation and do not predict GPU or neuromorphic performance.

For a hardware-independent secondary measure, one multiply-accumulate (MAC) counts as two FLOPs. Every other explicit scalar operation counts as one FLOP, and tanh evaluations are reported separately. With $S = 268,800$ weight MACs, $R = 68,096$ hidden-error MACs, $A = 522$ affine output units, $H = 512$ hidden units, and $P = 269,322$ parameters, BP uses $B(2S + R)$ MACs per batch and PC uses $B[3S + T(S + R)]$. Appendix C, Section C.3, reports exact per-epoch counts; Table 4 gives the corresponding 30-epoch totals.

4 Implementation checks

The checks support two conclusions: the implementation follows the intended PC equations, and in the controlled validation cases, PC updates near the stated stopping tolerance pointed in almost the same direction as BP updates while remaining smaller. These controlled checks do not establish convergence throughout training or explain the remaining accuracy gap.

Initial comparison code did not implement the intended supervised PC–BP correspondence, so the PC model was rebuilt around

Table 1: Width-256 weight-gradient comparisons at $T = 200$, averaged over ten seeds and all weight layers. Randomly initialized, untrained networks used one synthetic example per seed, target perturbation $\epsilon = 0.01$, and float64 arithmetic. “To PC limit” uses a separately converged PC gradient stopped at finite stationarity tolerance 10^{-8} ; “to BP” uses the matched BP gradient.

Depth	Cosine	Norm ratio	To BP	To PC limit
2	0.999	0.740	0.264	5.70×10^{-5}
4	0.998	0.681	0.322	1.36×10^{-4}
6	0.998	0.673	0.331	6.61×10^{-4}

the MSE objective. Before the learning comparison, the replacement was tested for matched parameter orientation and initialization, detached inferred beliefs, and synchronous hidden-layer updates. Validation used float64 arithmetic and raw gradients before any optimizer step. The convergence comparisons in Table 1 and Figure 1C–D used randomly initialized, untrained networks, one synthetic example per seed, and target perturbation $\epsilon = 0.01$. Their separately converged reference stopped at the finite stationarity tolerance of 10^{-8} . Appendix B and Table 5 give the numerical checks.

Manual activity gradients matched automatic differentiation, while the detached parameter gradients matched both automatic differentiation and centred finite differences. A separately written energy-based model reproduced the same beliefs, energy, stationarity, and gradients. These results establish internal correctness, not equivalence to BP.

BP comparisons used targets $y_\epsilon = o + \epsilon(y - o)$. For $\epsilon \leq 10^{-3}$ in depth-2 networks, the minimum mean weight-gradient cosine across activation, perturbation, and layer cells was 0.992; the overall mean was 0.994, and the smallest individual cosine was 0.970. Figure 1A–B shows that this strong directional agreement coexisted with PC/BP norm ratios below one: the updates pointed in similar directions, but PC’s updates remained smaller.

Figure 1 separates convergence to the PC fixed point from agreement with BP. At width 256 and $T = 200$, the finite-inference PC update was close to the PC update computed after inference reached the stated stopping tolerance; Table 1 gives the exact distances at depths two, four, and six. The small depth-two reversal after $T = 100$ is consistent with comparison against a finite-tolerance reference and should not be interpreted as a loss of inference convergence. The difference from BP stopped decreasing at a much larger value. At $T = 200$, cosines exceeded 0.998, whereas PC/BP norm ratios fell from 0.740 at depth two to 0.673 at depth six, showing that the residual difference was mainly in update size.

In the original benchmark, PC and BP shared the same truncated learning-rate grid and every MSE condition selected its upper boundary. With similarly directed gradients, a common nominal rate produced smaller PC parameter steps. The learning-rate extension instead selected each method independently and chose 0.04 for PC versus 0.03 for BP in the depth-2, 100-label MNIST condition. In a separate 100-example check, five epochs reduced feedforward

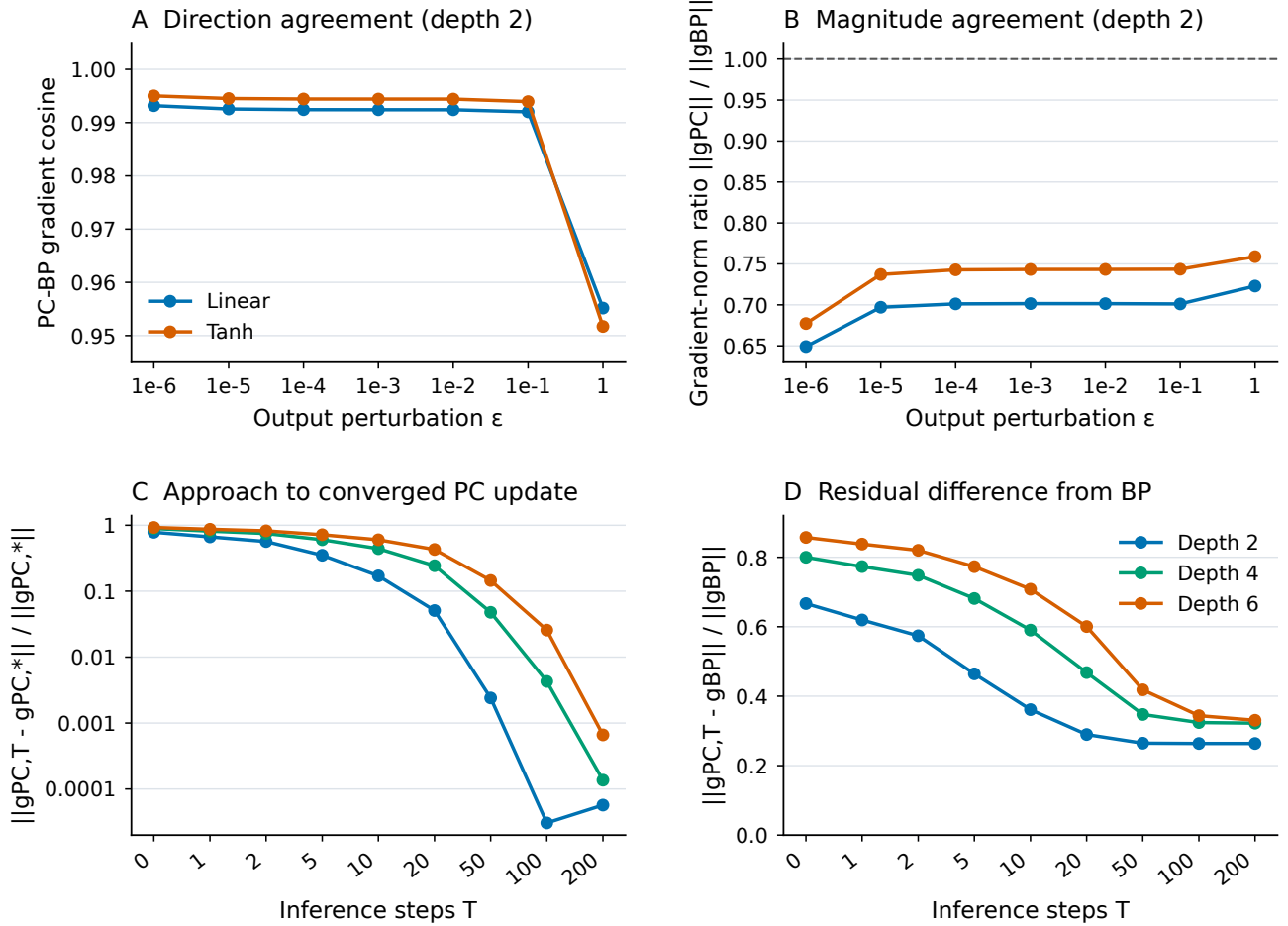


Figure 1: Gradient checks for unit-precision PC. (A–B) Depth-2 gradients remain closely aligned over small output perturbations, but their norm ratio stays below one. Values average ten seeds and three weight layers. (C) Finite-inference gradients generally approach separately converged PC gradients as T increases. The small depth-two reversal after $T = 100$ is consistent with comparison against a finite-tolerance reference and should not be interpreted as a loss of inference convergence. (D) Their difference from BP stops decreasing at a much larger value. Panels C–D use randomly initialized, untrained width-256 networks, one synthetic example per seed, target perturbation $\epsilon = 0.01$, float64 arithmetic, ten seeds, and all weight layers. The separately converged reference stopped at stationarity tolerance 10^{-8} . The two lower panels use different references and answer different questions.

MSE from 0.587 to 0.529 for BP and from 0.587 to 0.544 for PC, confirming that both implementations could learn.

5 Results

In the MNIST experiment with two hidden layers and 100 labelled training examples, PC came close to BP’s accuracy but required much more computation. With 200 inference steps, PC finished 30 epochs an average of 0.295 percentage points below BP and took about 131 times as long to train on a single CPU thread.

Each method’s learning rate was selected independently using validation data, and both MNIST searches met the prespecified bracketing rule. Fashion-MNIST remained unresolved because BP

did not show the required numerically valid decline in validation accuracy at a higher learning rate.

5.1 Validation-only learning-rate selection

Both depth-2 MNIST searches met the rule: validation accuracy was lower at numerically valid rates below and above each near-best group, and at least four of five paired seeds changed in the required direction. Figure 2 shows a performance zoom and, below it, the status of every fixed-grid candidate across the full range. Table 2 reports all four development cells, including Fashion-MNIST. Detailed candidate and failure evidence appears in Appendix C.

The Fashion-MNIST BP cell used all three permitted upper mid-point attempts (0.10954, 0.10466, and 0.10708). Near the numerical

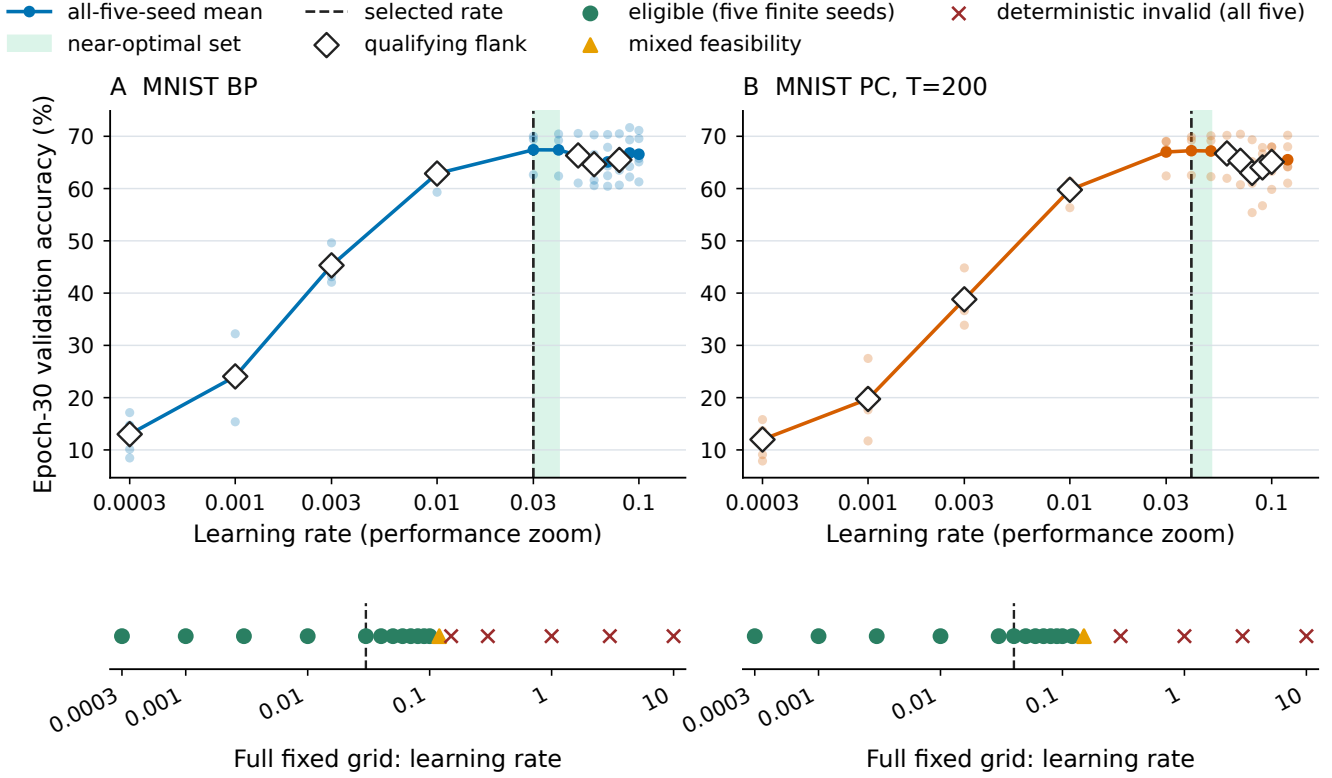


Figure 2: Depth-2 MNIST learning-rate evidence from the prospective extension. Each upper panel has its own labelled logarithmic performance-zoom axis; pale points are the five development seeds, connected points are eligible all-five-seed means, the shaded region is the near-optimal set, the dashed line is the selected rate, and open diamonds mark qualifying finite flanks. Each lower strip uses a separate full-grid logarithmic axis spanning 0.0003–10 and represents every one of the 18 fixed-grid candidates exactly once: green circles are eligible, amber triangles have mixed feasibility, and red crosses are deterministic invalid in all five seeds. Mixed and invalid candidates are not treated as performance turnover. Both method cells have lower and upper qualifying flanks.

Table 2: Validation-only selection in the four learning-rate extension cells. Flanks are qualifying finite rates under the prespecified 0.5-point mean downturn and four-of-five paired direction rule. Fashion-MNIST remains unresolved because its BP cell lacks an upper flank.

Dataset	Method	Selected	Near-optimal set	Qualifying lower flanks	Qualifying upper flanks
MNIST	BP	0.03	0.03, 0.04	0.0003, 0.001, 0.003, 0.01	0.05, 0.06, 0.08
MNIST	PC, $T = 200$	0.04	0.04, 0.05	0.0003, 0.001, 0.003, 0.01	0.06, 0.07, 0.08, 0.09, 0.1
Fashion-MNIST	BP	0.03	0.03, 0.1, 0.10466	0.0003, 0.001, 0.003, 0.01	none; unresolved
Fashion-MNIST	PC, $T = 200$	0.03	0.03	0.0003, 0.001, 0.003, 0.01	0.05, 0.06

boundary, some rates completed successfully in some seeds but failed in others. Because no numerically valid higher-rate candidate met the required downturn, the cell was fixed as unresolved before test access.

5.2 Accuracy after independent learning-rate selection

Figure 3 shows every paired confirmatory seed and the prespecified intervals. Table 3 gives the corresponding numerical summaries.

In the fully bracketed depth-2, 100-label MNIST comparison, PC at $T = 200$ minus BP at epoch 30 averaged -0.295 percentage points (median -0.320 , sample SD 0.172; 95% paired-bootstrap interval $[-0.376, -0.181]$). Nine of ten paired seeds were negative. The normalized presentation-AUC difference averaged -0.226 percentage points, with interval $[-0.291, -0.149]$.

Fashion-MNIST’s descriptive epoch-30 difference was -0.378 percentage points. The median was -0.390 , the SD was 0.162, and the interval was $[-0.475, -0.281]$. All ten seed differences were

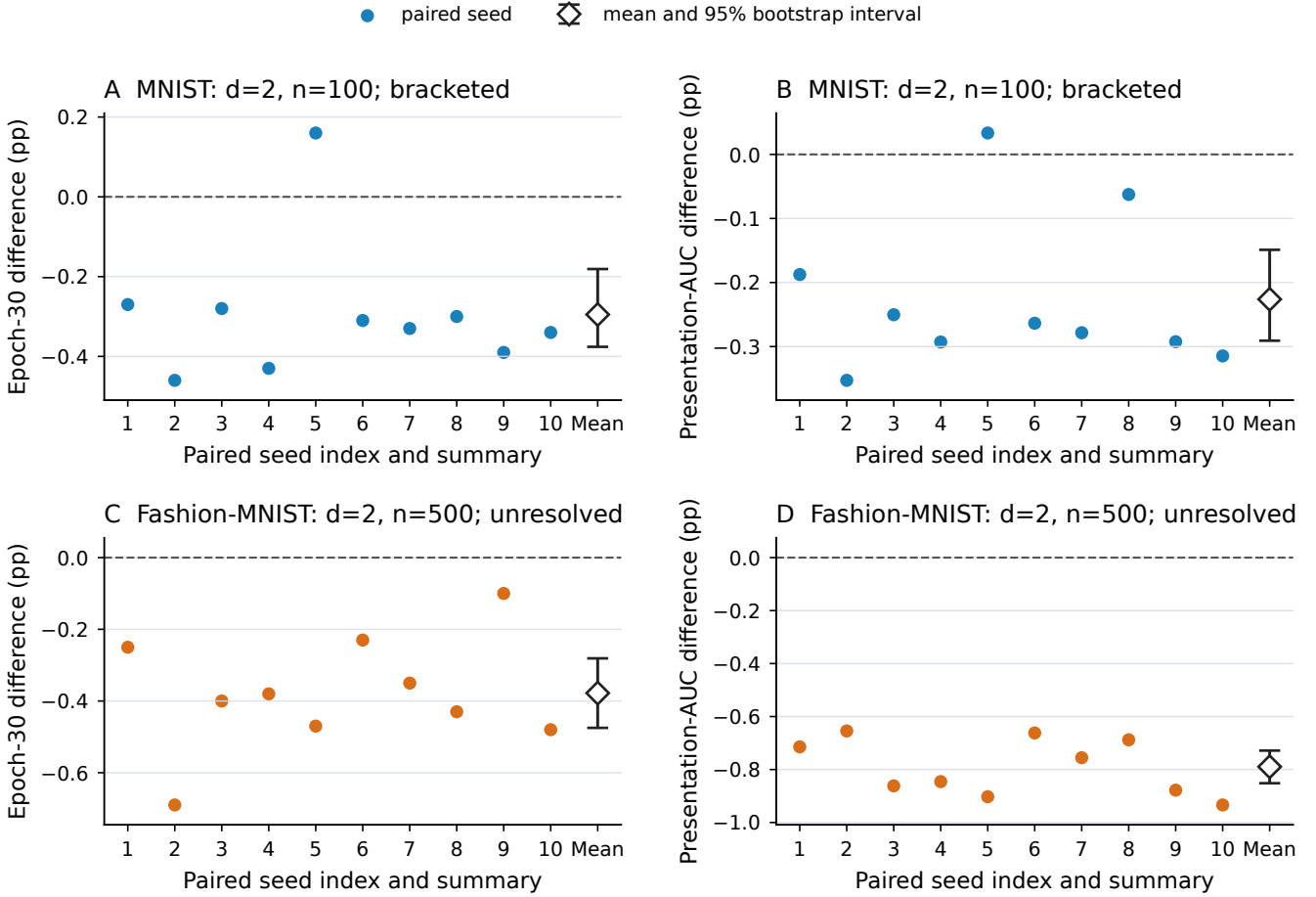


Figure 3: Paired accuracy results after independent learning-rate selection. Each circle is one of ten fresh paired seeds at positions 1–10; the open diamond and bar at the separate “Mean” position are the mean and 95% paired percentile-bootstrap interval from 10,000 seed-level resamples. The dashed horizontal line marks equality. The depth-2, 100-label MNIST method cells meet the prespecified bracketing rule. Fashion-MNIST is retained but labelled unresolved because BP did not show the required numerically valid decline at a higher rate.

Table 3: Paired summaries after independent learning-rate selection. Differences are PC minus BP in percentage points. Time is the geometric mean paired ratio. Fashion-MNIST is descriptive because its search is unresolved.

Quantity	MNIST ($d = 2, n = 100$)	Fashion-MNIST ($d = 2, n = 500$)
Search status	bracketed	unresolved
Epoch-30 mean	-0.295	-0.378
Epoch-30 median	-0.320	-0.390
Epoch-30 SD	0.172	0.162
Epoch-30 95% interval	$[-0.376, -0.181]$	$[-0.475, -0.281]$
AUC mean	-0.226	-0.790
AUC 95% interval	$[-0.291, -0.149]$	$[-0.852, -0.729]$
PC/BP time ratio	130.569	177.153
Time-ratio 95% interval	$[129.155, 131.788]$	$[175.817, 178.381]$

negative. Its AUC difference averaged -0.790 percentage points,

with interval $[-0.852, -0.729]$. These values describe the prespecified retained rates; they do not establish performance at bracketed optima.

5.3 Diagnostic-free time and analytical work

All confirmatory timings were float32, one-thread CPU measurements. PC’s geometric-mean diagnostic-free training-time ratio was 130.569 on MNIST (interval $[129.155, 131.788]$). On unresolved Fashion-MNIST it was 177.153 (interval $[175.817, 178.381]$). Every seed-level ratio exceeded 125 on MNIST and 172 on Fashion-MNIST. Unlike the original benchmark timing, these windows exclude PC-only energy, stationarity, and other diagnostic calculations.

Table 4 reports the hardware-independent operation counts under the stated convention. PC at $T = 200$ used 112.574 times the BP MACs on both datasets, 112.035 times the FLOPs on MNIST and 112.229 times on Fashion-MNIST, and 403 times the tanh evaluations. In rounded terms, these are approximately 113 times as

many MACs, 112 times as many FLOPs, and 403 times as many tanh evaluations. We did not measure GPU performance and make no claim about its speed ratio. A matched GPU comparison remains future work.

5.4 Original broad benchmark as separate evidence

The original 48-condition benchmark is not replaced by the extension. Figure 4 retains its endpoint results as a separate descriptive experiment. All 48 mean PC-minus-BP effects were negative, 47 intervals were wholly negative, and the median effect was -2.92 percentage points. The grid was unbracketed, and PC accuracy was still increasing with inference budget in 15 of 16 cells, so these summaries do not represent globally tuned comparisons.

For the same depth-2 architecture and 100-label MNIST set size, the original unbracketed mean at rate 0.03 and PC $T = 50$ was -0.786 percentage points. In the separate extension, the mean at independently selected rates and PC $T = 200$ was 0.491 points closer to zero. The searches, inference budgets, and seed sets differ, so this is neither a formal cross-study test nor evidence that any one design change caused the difference.

The original instrumented-time ratios of 7.96–67.3 are retained only as historical context because PC-only diagnostics were inside the PC timing window. The learning-rate extension’s diagnostic-free ratios are the primary timing evidence for the extended-search comparison.

6 Discussion and limitations

The learning-rate extension strengthens the depth-2, 100-label MNIST comparison without establishing a global optimum. Both method searches found numerically valid declines on either side of the near-best rates, and ten untouched confirmatory seeds showed a mean PC disadvantage of 0.295 percentage points at epoch 30 with PC at $T = 200$. For the same architecture and labelled-set size, the original unbracketed mean was -0.786 points; the gap measured in the extension was 0.491 points smaller. The learning-rate searches, inference budgets, and seed sets differed, so this descriptive comparison does not identify which change produced the difference. Neither the epoch-30 result nor the presentation-AUC result reached parity.

Diagnostic-free timing shows a large computational difference alongside the small MNIST accuracy difference. On the same float32, single-thread CPU setup, PC at $T = 200$ took about 131 times BP’s training time on MNIST and 177 times on Fashion-MNIST. Warm-up, evaluation, diagnostics, serialization, and logging were outside both timed totals. Analytical counts point in the same direction: approximately 113 times as many MACs, 112 times as many FLOPs, and 403 times as many tanh evaluations. These values characterize this implementation and architecture. They do not predict GPU, analogue, or neuromorphic performance, where operation fusion, parallelism, memory movement, or physical dynamics may differ.

The controlled gradient checks help interpret, but do not explain, the accuracy result. PC updates computed after inference reached the stated stopping tolerance pointed in almost the same direction as BP updates while remaining smaller, with PC/BP norm ratios of 0.673–0.740 at width 256. The checks do not establish convergence

throughout training or identify the cause of the remaining gap. The small depth-two reversal beyond $T = 100$ is consistent with comparison against a finite-tolerance reference; it should not be interpreted as a loss of inference convergence.

Fashion-MNIST must be interpreted differently. Its PC search met the bracketing rule, but near the upper numerical boundary the BP runs sometimes succeeded and sometimes failed, and no numerically valid higher rate showed the required decline. The selected BP rate completed successfully in all five development seeds, so the confirmatory comparison describes the retained rates; it is not evidence from jointly bracketed optima. Keeping this unresolved result visible avoids mistaking numerical instability for a decline in validation performance.

The broader empirical scope is limited: two small grayscale datasets, width-256 tanh networks, depth two in the extension, 30 epochs, MSE and plain SGD, no convolution, normalization layers, residual connections, or pretraining, and one synchronous unit-precision PC formulation. The original benchmark covers 48 depth, labelled-set-size, and inference-budget comparisons, but its learning-rate search was truncated and its timing included PC-only diagnostics; its evidence remains separate from the extension. Ten paired confirmatory seeds quantify variation in these cells but do not support universal claims. The bootstrap intervals are conditional on the selected rates, fixed split, and confirmatory design; they do not incorporate uncertainty from repeating the development search and are not presented as significance tests.

Precision weighting and fixed predictions remain important future directions. A comparison with such a variant would help determine whether the observed deficit changes under that intervention; the present study does not establish that specificity. Likewise, GPU measurements would be valuable, but were not made here. Future comparisons should state the variant and matching rules clearly and use matched implementations designed for the target hardware.

At the independently selected rates and $T = 200$, the tested depth-2, 100-label MNIST condition therefore retained a small accuracy disadvantage and substantially greater computational cost. The study does not establish a global optimum over inference budgets or exclude incomplete inference as a contributor during training. Other predictive-coding algorithms and hardware remain open empirical questions.

7 Conclusion

In a two-hidden-layer MNIST network trained on 100 labelled examples, synchronous unit-precision PC with 200 inference steps was 0.295 percentage points less accurate than BP after 30 epochs, despite both learning-rate searches meeting the prespecified bracketing rule. Training took about 131 times as long on a single CPU thread. These results show a small accuracy disadvantage and a large computational cost for the tested configuration. Fashion-MNIST remained unresolved. The validation checks do not establish convergence throughout training or explain the accuracy gap, and the findings do not rank other PC variants or hardware.

References

- [1] Francis Crick. 1989. The recent excitement about neural networks. *Nature* 337, 6203 (1989), 129–132. <https://doi.org/10.1038/337129a0>

Table 4: Analytical operation counts for one 30-epoch confirmatory run. One MAC equals two FLOPs; other explicit scalar operations contribute one FLOP, and tanh evaluations are counted separately.

Dataset	Method	30-epoch MACs	30-epoch FLOPs	30-epoch tanh evaluations
MNIST, $n = 100$	BP	1,817,088,000	3,674,323,380	1,536,000
MNIST, $n = 100$	PC, $T = 200$	204,556,800,000	411,654,069,960	619,008,000
Fashion-MNIST, $n = 500$	BP	9,085,440,000	18,339,329,520	7,680,000
Fashion-MNIST, $n = 500$	PC, $T = 200$	1,022,784,000,000	2,058,205,743,840	3,095,040,000

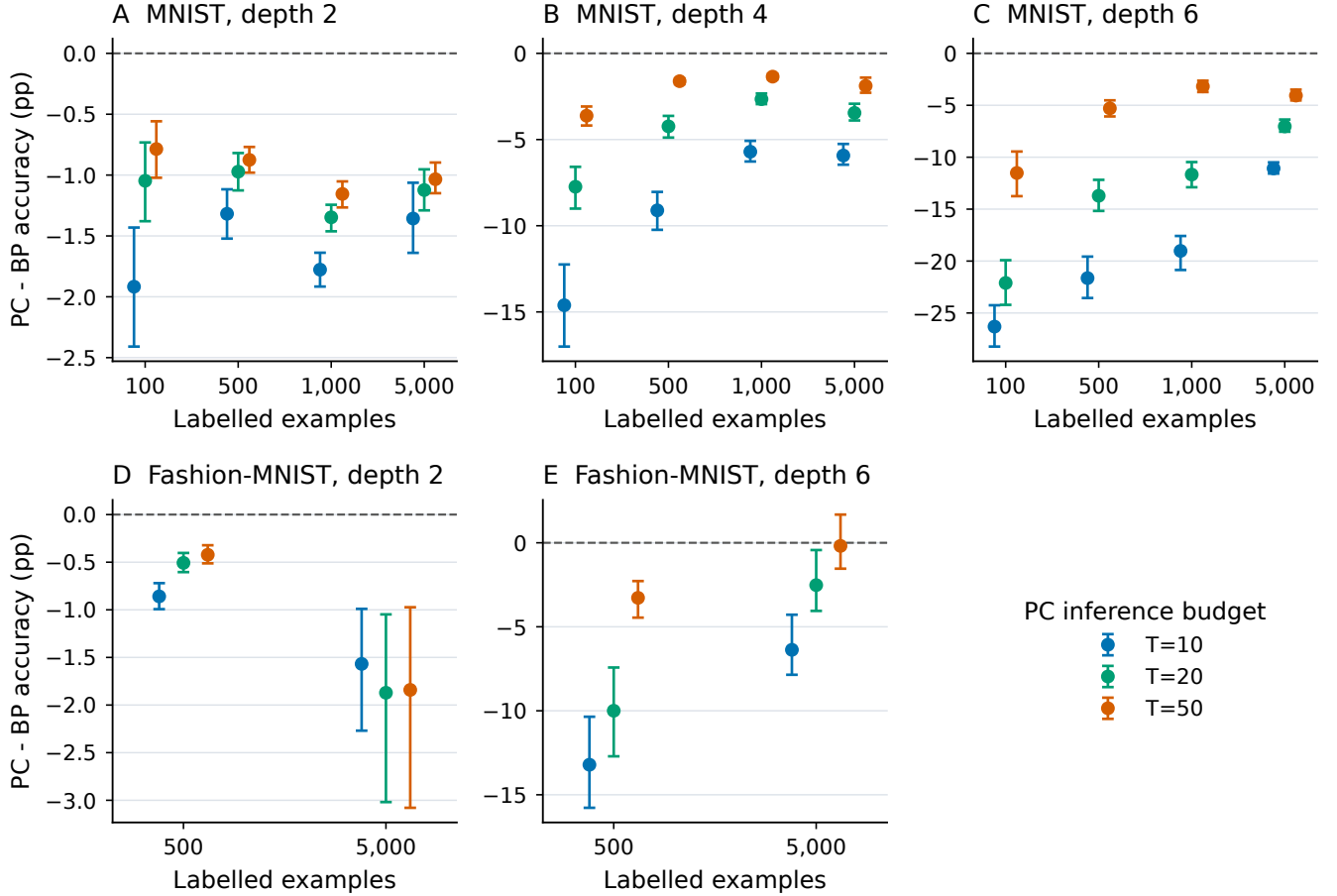


Figure 4: Original broad benchmark, reported separately from the learning-rate extension. Each point is a mean PC-minus-BP epoch-30 difference across ten paired seeds; bars are 95% paired percentile-bootstrap intervals. The original six-rate learning-rate grid ended at 0.03 before validation turnover, so this figure is descriptive evidence from an unbracketed search. No original seed is pooled with the extension.

- [2] Simon Frieder and Thomas Lukasiewicz. 2022. (Non-)Convergence Results for Predictive Coding Networks. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*. PMLR, 6793–6810. <https://proceedings.mlr.press/v162/frieder22a.html>
- [3] Cédric Goemaere, Gaspard Oliviers, Rafal Bogacz, and Thomas Demeester. 2026. ePC: Fast and Deep Predictive Coding in Digital Simulation. <https://doi.org/10.48550/arXiv.2505.20137> arXiv:2505.20137 [cs.LG] Version 5, revised 8 June 2026; accepted to the ICML 2026 main track.
- [4] Francesco Innocenti, El Mehdi Achour, and Christopher L. Buckley. 2025. μ PC: Scaling Predictive Coding to 100+ Layer Networks. In *Advances in Neural Information Processing Systems 38*. <https://doi.org/10.52202/085713-2923>
- [5] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations*. San Diego, California, USA. <https://arxiv.org/abs/1412.6980>
- [6] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324. <https://doi.org/10.1109/5.726791>
- [7] Beren Millidge, Alexander Tschantz, and Christopher L. Buckley. 2022. Predictive Coding Approximates Backprop Along Arbitrary Computation Graphs. *Neural Computation* 34, 6 (2022), 1329–1368. https://doi.org/10.1162/neco_a_01497
- [8] Gaspard Oliviers, Elene Lominadze, and Rafal Bogacz. 2026. Understanding Sample Efficiency in Predictive Coding. <https://doi.org/10.48550/arXiv.2605>

11911 arXiv:2605.11911 [cs.LG]

- [9] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems* 32, Hanna Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché Buc, Emily Fox, and Roman Garnett (Eds.). Curran Associates, Inc., 8024–8035. <https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html>
- [10] Luca Pinchetti, Chang Qi, Oleh Lokshyn, Cornelius Emde, Amine M'Charrak, Mufeng Tang, Simon Frieder, Bayar Menzat, Gaspard Oliviers, Rafal Bogacz, Thomas Lukasiewicz, and Tommaso Salvatori. 2025. Benchmarking Predictive Coding Networks – Made Simple. In *International Conference on Learning Representations*. https://proceedings.iclr.cc/paper_files/paper/2025/hash/581df42e8ebbecac39aeda03519b7c0e-Abstract-Conference.html
- [11] Rajesh P. N. Rao and Dana H. Ballard. 1999. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience* 2, 1 (1999), 79–87. <https://doi.org/10.1038/4580>
- [12] Robert Rosenbaum. 2022. On the relationship between predictive coding and backpropagation. *PLOS ONE* 17, 3 (2022), e0266102. <https://doi.org/10.1371/journal.pone.0266102>
- [13] Robert Rosenbaum. 2025. Correction: On the relationship between predictive coding and backpropagation. *PLOS ONE* 20, 3 (2025), e0320944. <https://doi.org/10.1371/journal.pone.0320944>
- [14] Tommaso Salvatori, Yuhang Song, Zhenghua Xu, Thomas Lukasiewicz, and Rafal Bogacz. 2022. Reverse Differentiation via Predictive Coding. *Proceedings of the AAAI Conference on Artificial Intelligence* 36, 7 (2022), 8150–8158. <https://doi.org/10.1609/aaai.v36i7.20788>
- [15] Tommaso Salvatori, Yuhang Song, Yordan Yordanov, Beren Millidge, Lei Sha, Cornelius Emde, Zhenghua Xu, Rafal Bogacz, and Thomas Lukasiewicz. 2024. A Stable, Fast, and Fully Automatic Learning Algorithm for Predictive Coding Networks. In *International Conference on Learning Representations*. https://proceedings.iclr.cc/paper_files/paper/2024/hash/554414e570a85eb3118e988c5d77986f-Abstract-Conference.html
- [16] Alexander Tschantz, Beren Millidge, Anil K. Seth, and Christopher L. Buckley. 2023. Hybrid predictive coding: Inferring, fast and slow. *PLOS Computational Biology* 19, 8 (2023), e1011280. <https://doi.org/10.1371/journal.pcbi.1011280>
- [17] James C. R. Whittington and Rafal Bogacz. 2017. An Approximation of the Error Backpropagation Algorithm in a Predictive Coding Network with Local Hebbian Synaptic Plasticity. *Neural Computation* 29, 5 (2017), 1229–1262. https://doi.org/10.1162/NECO_a_00949
- [18] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. <https://doi.org/10.48550/arXiv.1708.07747> arXiv:1708.07747 [cs.LG]
- [19] Umais Zahid, Qinghai Guo, and Zafeirios Fountas. 2023. Predictive Coding as a Neuromorphic Alternative to Backpropagation: A Critical Evaluation. *Neural Computation* 35, 12 (2023), 1881–1909. https://doi.org/10.1162/neco_a_01620

A Algorithmic derivatives and numerical conventions

For hidden layer l , differentiating the summed energy in Equation (2) gives

$$\frac{\partial E_{\text{sum}}}{\partial v_{b,l}} = \epsilon_{b,l} - (\epsilon_{b,l+1} W_l) \odot \phi'_l(v_{b,l}). \quad (5)$$

All hidden layers use Equation (5) from the same pre-update beliefs at each inference step. Input and target output remain clamped. Hidden predictions are $\tanh(v_l)$; input activity is the identity and output activity is linear.

With inferred beliefs detached, the bias derivative accompanying Equation (4) is

$$\frac{\partial E_{\text{mean}}}{\partial b_l} = -\frac{1}{B} \sum_{b=1}^B \epsilon_{b,l+1}. \quad (6)$$

The stored weight orientation is output-by-input, so prediction uses $\phi_l(v_l) W_l^\top$. Activity inference follows the sum-energy derivative, which preserves the per-example trajectory under batch vectorization. Parameter learning follows mean energy, matching BP's mean-per-example MSE scaling. Differentiating through the inference path would define a different learning rule.

Stationarity is the maximum, over hidden layers, of the root-mean-square entries of $\partial E_{\text{sum}} / \partial v_l$. Relative L2 discrepancy between a candidate gradient g and reference g_r is $\|g - g_r\|_2 / \|g_r\|_2$. The gradient-norm ratio is $\|g_{\text{PC}}\|_2 / \|g_{\text{BP}}\|_2$.

Validation used float64 arithmetic and raw gradients before any optimizer step. Training and confirmatory timing used float32. These precision settings serve different purposes: float64 exposes derivative and convergence discrepancies, while float32 matches the training implementation.

B Mathematical and numerical checks

Table 5: Checks completed before the learning comparison. The directional criterion in Check 6 required a minimum layer/activation/perturbation mean cosine of 0.95 and an across-cell mean of 0.98 for $\epsilon \leq 10^{-3}$.

Check	Purpose	Result
1	Feedforward identity	BP and PC activities agreed at every layer after exact parameter matching.
2	Zero-error behaviour	A feedforward target produced negligible energy, gradients, and belief movement.
3	Activity derivatives	Manual $\nabla_v E_{\text{sum}}$ matched automatic differentiation for batches of one and eight.
4	Parameter derivatives	Detached-belief gradients of E_{mean} matched automatic differentiation and centred finite differences.
5	Inference stability	Accepted rates remained finite, decreased energy, and improved stationarity.
6	Small-error direction	The minimum mean cosine was 0.992 and the across-cell mean was 0.994; magnitude discrepancies remained.
7	Inference budget	Finite-budget PC gradients approached separately converged PC gradients as stationarity improved.
8	Independent formulation	A separately written energy model reproduced beliefs, gradients, energy, and stationarity.
9	Learning check	BP-MSE and PC-MSE both reduced MSE on the same 100-example subset.

For a batch of size B , the parameter derivatives of E_{mean} are

$$\frac{\partial E_{\text{mean}}}{\partial W_l} = -\frac{1}{B} \sum_{b=1}^B \epsilon_{b,l+1}^\top \phi_l(v_{b,l}), \quad (7)$$

$$\frac{\partial E_{\text{mean}}}{\partial b_l} = -\frac{1}{B} \sum_{b=1}^B \epsilon_{b,l+1}. \quad (8)$$

The beliefs on the right-hand side are detached. Differentiating through the inference trajectory would define a different learning rule. Conversely, using the gradient of mean energy for activity inference without compensating for B would shrink each belief update by $1/B$ and make the dynamics depend on vectorized batch size.

The converged comparisons used an absolute stationarity criterion of 10^{-8} . Since stationarity is defined from E_{sum} , its numerical scale depends on the batch convention; the validation kept that convention explicit and separately verified batch-invariant trajectories. Small output perturbations also produce small gradient norms, so absolute discrepancies can vanish while relative discrepancies remain appreciable.

All BP comparisons used the MSE objective, raw pre-optimizer gradients, matched parameter orientation and initialization, detached inferred beliefs, synchronous activity updates, and float64 validation arithmetic. These checks concern unit-precision PC without an output-precision hierarchy or fixed-prediction condition.

C Extended-search methods and results

C.1 Development evidence and retained failures

All four cells completed the same 18-rate fixed grid before any adaptive action. The eligible and ineligible sets are shown in Table 6. A candidate was eligible only if all five development seeds completed. Across the 375 candidate-seed identities, 102 deterministic-invalid trajectories were retained: 58 ‘nonfinite_parameter’ and 44 ‘observed_energy_increase’. Finite seeds from mixed candidates were never averaged for selection.

Table 6: Rate-level development status. Rates not listed as ineligible were eligible in all five seeds. Fashion-MNIST BP includes its three adaptive midpoint candidates.

Cell	Eligible rates	Ineligible or mixed rates
MNIST BP	0.0003–0.1 on the fixed grid	0.12, 0.15, 0.3, 1, 3, 10
MNIST PC, $T = 200$	0.0003–0.12 on the fixed grid	0.15, 0.3, 1, 3, 10
Fashion-MNIST BP	0.0003–0.1; 0.1046635; 0.1070762	0.1095445; 0.12, 0.15, 0.3, 1, 3, 10
Fashion-MNIST PC, $T = 200$	0.0003–0.12 on the fixed grid	0.15, 0.3, 1, 3, 10

The notation “0.0003–0.1 on the fixed grid” refers to discrete tested values, not a continuous interval. Those values were 0.0003, 0.001, 0.003, 0.01, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, and 0.1. MNIST PC additionally retained 0.12 as eligible. Fashion-MNIST BP remained unresolved because no eligible upper rate met the finite downturn rule after all three midpoint attempts; mixed feasibility and numerical failure were not counted as turnover.

The earlier v1.1 and v1.2 construction attempts were separate protocol-development exercises with development seeds 100–102 and 200–204. They did not share the extension’s seed namespace, did not enter its selection means, and were not pooled with confirmatory outcomes. They document method development rather than evidence for the reported effect estimates.

C.2 Seed-level confirmatory summaries

Table 7 reports the paired values plotted in Figure 3. Training seconds sum only the 30 diagnostic-free timed epochs; disposable warm-ups are excluded.

Table 7: Seed-level extended-search results. Δ_{30} and Δ_{AUC} are PC minus BP in percentage points. Fashion-MNIST remains unresolved.

Dataset/seed	Δ_{30}	Δ_{AUC}	BP seconds	PC seconds	PC/BP
MNIST 41001	−0.27	−0.188	0.023215	3.006470	129.505
MNIST 41002	−0.46	−0.353	0.022788	3.025790	132.778
MNIST 41003	−0.28	−0.250	0.023931	3.000345	125.377
MNIST 41004	−0.43	−0.293	0.022689	3.022946	133.235
MNIST 41005	+0.16	+0.033	0.022970	2.990691	130.200
MNIST 41006	−0.31	−0.264	0.023200	3.020499	130.196
MNIST 41007	−0.33	−0.278	0.022685	2.995503	132.050
MNIST 41008	−0.30	−0.063	0.022982	2.990836	130.138
MNIST 41009	−0.39	−0.292	0.023154	3.011018	130.044
MNIST 41010	−0.34	−0.315	0.022702	3.004529	132.345
Fashion-MNIST 41001	−0.25	−0.714	0.076434	13.367747	174.893
Fashion-MNIST 41002	−0.69	−0.654	0.075792	13.356050	176.219
Fashion-MNIST 41003	−0.40	−0.862	0.075232	13.375387	177.787
Fashion-MNIST 41004	−0.38	−0.846	0.075068	13.378833	178.222
Fashion-MNIST 41005	−0.47	−0.903	0.074678	13.407663	179.539
Fashion-MNIST 41006	−0.23	−0.662	0.077229	13.351318	172.879
Fashion-MNIST 41007	−0.35	−0.755	0.075904	13.395730	176.482
Fashion-MNIST 41008	−0.43	−0.688	0.075641	13.408735	177.267
Fashion-MNIST 41009	−0.10	−0.878	0.074162	13.364089	180.202
Fashion-MNIST 41010	−0.48	−0.934	0.075281	13.412106	178.160

C.3 Per-epoch operation counts

For the fixed architecture, $S = 268,800$ is the number of weight MACs per forward example, $R = 68,096$ the hidden-error MACs per example, $A = 522$ the affine output units, $H = 512$ the hidden units, and $P = 269,322$ the trainable parameters. For batch size B , BP uses $B(2S + R)$ MACs and PC uses $B[3S + T(S + R)]$ MACs. Table 8 gives the exact per-epoch counts underlying Table 4.

Table 8: Exact analytical counts per epoch. One MAC is two FLOPs; other explicit scalar operations count as one FLOP, and tanh evaluations are separate.

Dataset	Method	MACs	FLOPs	tanh evaluations
MNIST	BP	60,569,600	122,477,446	51,200
MNIST	PC, $T = 200$	6,818,560,000	13,721,802,332	20,633,600
Fashion-MNIST	BP	302,848,000	611,310,984	256,000
Fashion-MNIST	PC, $T = 200$	34,092,800,000	68,606,858,128	103,168,000